ฉบับที่ 8

ไมโคร อิเลคทรอนิคส์

ปีที่ 1 / 2527

วารสารไมโครคอมพิวเตอร์ฉบับแรกในประเทศไทย

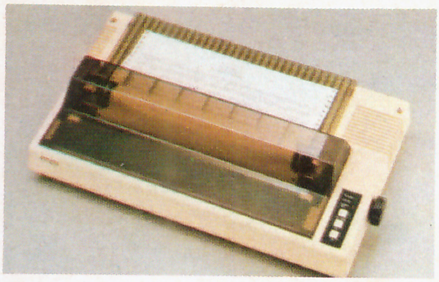
- ผ่า IBM PC
- รหัสอักษรภาษาไทย
- เผยโฉม PC jr และ Mcintosh
- คู่มือ MBASIC
- พูดจาภาษา BASIC



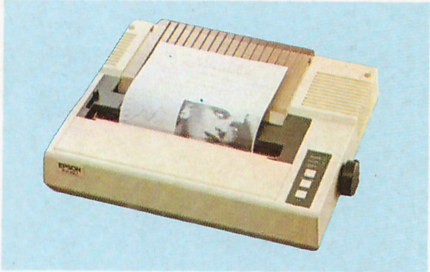
20.

EPSON : Last Word in Technology

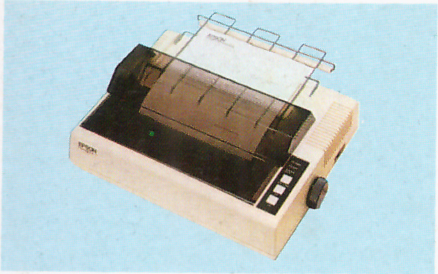
SIC



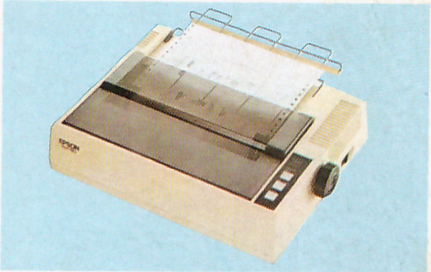
Dynamic Printer FX-100
160 cps printing speed
136 columns per line (normal mode)
Tractor/friction paper feed



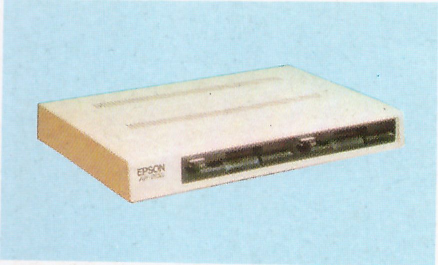
Versatile Printer FX-80
160 cps printing speed
80 columns per line (normal mode)
Tractor/friction paper feed
(cut sheet or roll paper)



Personal Printer with Friction Feed RX-80 F/T
100 cps printing speed
80 columns per line (normal mode)
Tractor/friction paper feed (cut sheet or roll paper)



Personal Printer RX-80
100 cps printing speed
80 columns per line (normal mode)
Tractor paper feed



AP-200 Ultraslim Floppy Disk Unit
100% Compatible with Apple II and Apple II +
Single - side 5¼" Floppy Disk Drive
143 KB (formatted) per drive or 286KB total



Portable Computer HX-20
RAM 16KB (expandable to 32KB)
ROM 32KB (expandable to 72KB)
Built-in Peripherals: Display, Printer, Clock, Music Generator, RS232C Serial Interface

SIC

บริษัท สหวิริยาอินเตอร์แนชชั่นแนลคอมพิวเตอร์ จำกัด
28/1 ตึกประภาวิทย์ ชั้น 3 ถนนสุรศักดิ์ สีลม กรุงเทพมหานคร
☎ 235-1887, 233-9878-9

Scotch® Diskettes, for record reliability.



Saving time and money ...the inside story:

- **Polyvinyl (PVC) jacket** protects against handling damage and provides crack resistance over an operating range of 50°-125° F.
- **Fabric Liner** helps keep disk free of dust and other contaminants.
- **Low modulation** provides better signal stability and reduced peak shift for concise, reliable performance.
- **Smooth surface** means better head-to-disk contact for uniform signal output. 3M diskettes measure less than 3 microinches.
- **Increased durability** means long-term reliability under heavy use.
- **Low abrasivity** extends the life of read/write heads for longer performance, reliability, and lower operating costs.
- **Stress relief notches** on the bottom edge help prevent creases and resist damage, even after repeated handling.



You can trust the quality and consistency of Scotch brand diskette. Each diskette is guaranteed against manufacturing defects. The care we take pays off for you in dependable performance, time after time, job after job.

Everything we put into quality Scotch brand diskettes means consistent performance. You won't trust anything less. If it's worth remembering, it's worth Scotch brand diskettes.

FOR MORE INFORMATION CALLS :

ACTION COMPUTER CO., LTD.
Soi Saladaeng Tel. 234-1577

AUTOMATION SERVICE CO., LTD.
New Petchburi Rd. Tel. 252-3210

BUSINESS MANAGEMENT COMPUTER CO., LTD.
Sukhumvit 30 A Tel. 391-3618

COMPUMATION CO., LTD.
Surawong Rd. Tel. 235-0200

COMPUTER TECHNOLOGY CO., LTD.
Vipawadee-Rangsit Rd. Tel. 279-4892

DATAMAT CO., LTD.
New Petchburi Rd. Tel. 314-5451-6

GRUNO (THAILAND) CO., LTD.
4th Floor Siam Center Tel. 251-9615

INNOVATION CORPORATION
4th Floor Siam Center Tel. 251-3902

INTER PROFESSIONAL COMPUTER CO., LTD.
Ploenchit Rd. Tel. 252-4918

KRUNGTHEP VITHYA
Rama I Rd. Tel. 251-3495

LOXLEY BANGKOK CO., LTD.
Suapah Rd. Tel. 221-9156-60

SAHAVIRIYA INFORTECH COMPUTER CO., LTD.
Surasak Rd. Tel. 235-1387

SCT CO., LTD.
Phaholyothin Rd. Tel. 279-6509

UNICOMP INTERNATIONAL CO., LTD.
New Petchburi Rd. Tel. 252-4254-5

S.P. HAADYAI LP.
Saengchan Rd. Haadyai Songkla. Tel. 246017

3M Thailand Limited
1643/5 Petchburi Road Ext.
G.P.O. Box 1650 Bangkok.
Tel. 2519191-5 Cable: TRIMINCO-BANGKOK

3M Hears you...

3M

ไมโคร อิเลคทรอนิคส์

ฉบับที่ 8

วิทยาการคอมพิวเตอร์

โปรแกรมสำเร็จรูป

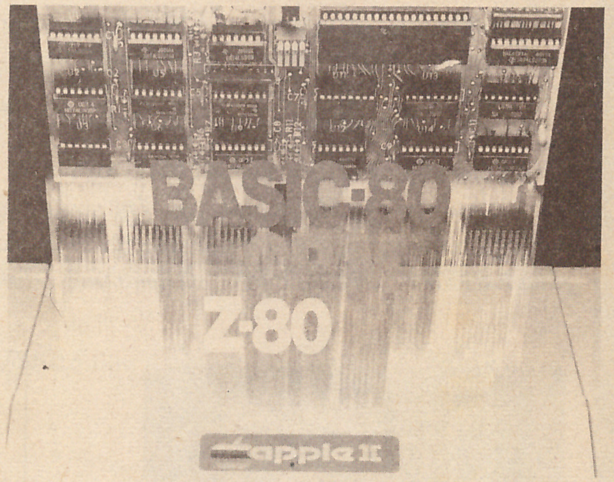
คอมพิวเตอร์ฮาร์ดแวร์

คอมพิวเตอร์ซอฟท์แวร์

ประจำฉบับ



43



102

หากคุณต้องการ CRT. MONITOR ที่ดีและราคาถูก สำหรับ VIDEO TAPE และ COMPUTER

อ่านตรงนี้

เราเป็นแห่งแรกที่รับดัดแปลง T.V. ของท่านทุกระบบให้เป็น CRT. MONITOR ระบบเดียวกับ COMPUTER ของท่าน เช่น APPLE II, IBM, COMMODORE, ATARI, SINCLAIR และอื่น ๆ **โดยไม่ต้องใช้ PAL CARD** เพราะการไม่ใช้ PAL CARD จะให้สีที่ดีกว่าเดิมมาก



-คุณจะเห็นคุณภาพที่ดีกว่าเดิมอย่างชัดเจนซึ่งท่านสามารถพิสูจน์ได้

-ด้วยค่าใช้จ่ายเพียงเล็กน้อย T.V. ของท่าน ก็สามารถเป็น T.V ได้ทุกระบบ และยังเป็น MONITOR ทุกระบบอีกด้วย นอกจากจะใช้กับ COMPUTER แล้ว ท่านยังสามารถใช้เป็น MONITOR เพื่อเล่นกับ VIDEO TAPE ได้อีกด้วย และยังเป็น T.V. ได้ตาม ปรกติเหมือนเดิมทุกประการ

-ทางร้านยังจำหน่าย :COMPUTER AP II, DISK DRIVE, PERIPHERAL CARD ต่าง ๆ, อะไหล่ COMPUTER, อุปกรณ์ทางด้าน HARDWARE และ SOFT WARE และตำราเกี่ยวกับ COMPUTER อีกมากมาย

-รับซ่อม : COMPUTER AP II หรือเครื่องเลียนแบบทุกชนิด, ซ่อม MONITOR ทุกยี่ห้อทั้ง GREEN และ COLOR MONITOR, VIDEO TAPE, T.V.สี

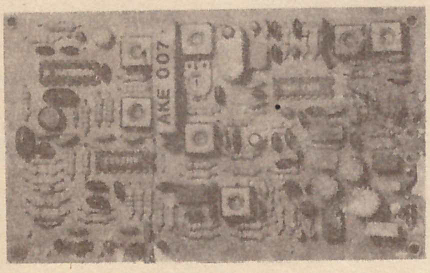
AKE
AKE ELECTONICS

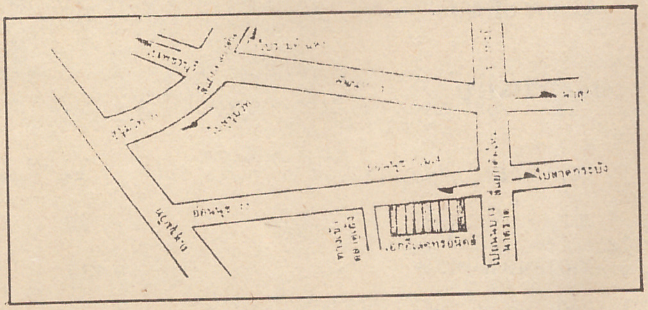
เอก อิ๊เลคทรอนิคส์

86/18-19 ซอยล่อนนุย 1 ปากทางเข้าตลาดเลื่อนฤาชัย
แขวงสวนหลวง เขตพระโขนง กทม. 10250 โทร.311-3035

-เราเป็นแห่งเดียวที่รับดัดแปลง T.V. สีทุก ระบบทั่วโลก เพื่อให้ใช้ได้ในประเทศไทย และยังจำหน่ายแผงสี AKE 007 เพื่อใช้ในการ แปลงระบบและซ่อม T.V. สี ซึ่งใช้ง่ายและ ได้ผลมากกว่าแผงสีทุกชนิดที่มีจำหน่ายใน ปัจจุบัน มีจำหน่ายที่ :

-ร้านธวัชชัยอะไหล่ -ทิพย์อีเล็คโทรนิคส์ -หนึ่งอีเล็คโทรนิคส์ -ไทยอีเล็คโทรนิคส์ -บ้านหม้ออีเล็คโทรนิคส์ -และร้านย่านบ้านหม้อทั่วไป

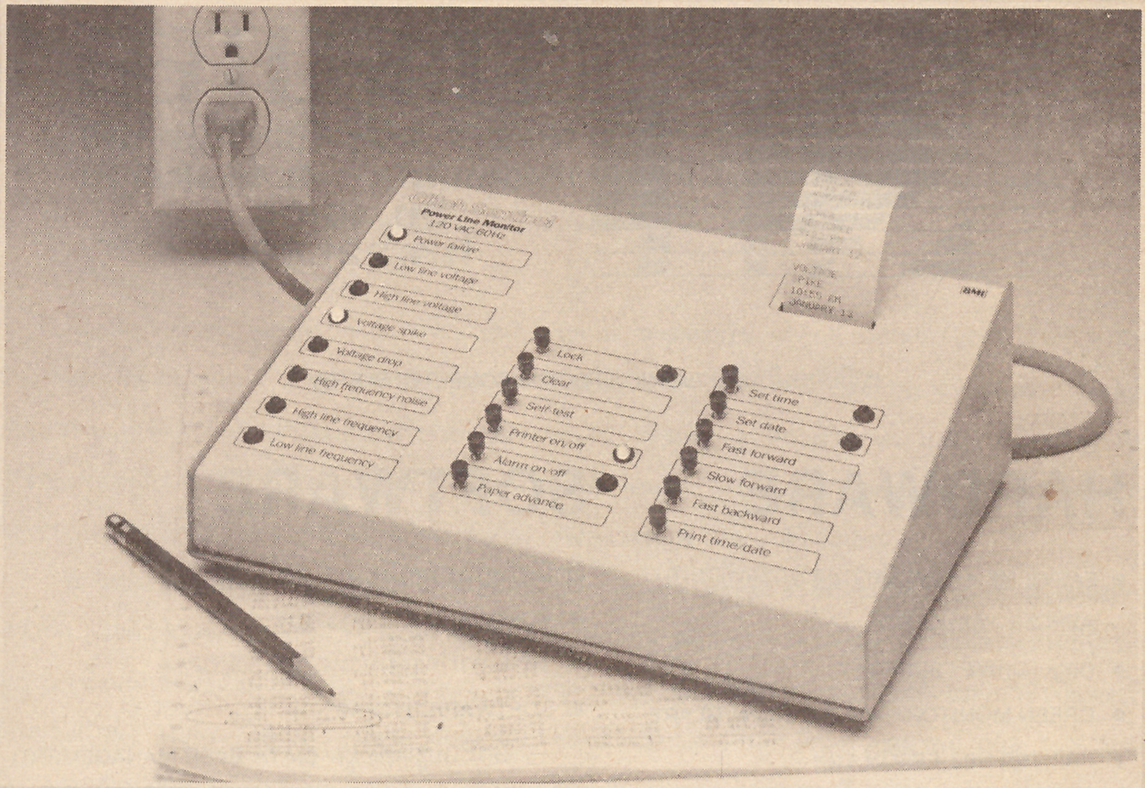






Concentrate. It's still cheaper!
Concentrate. It's still cheaper!
Concentrate. It's still cheaper!
Concentrate. It's still cheaper!
Concentrate. It's still cheaper!
Concentrate. It's still cheaper!
micom MicroComputers for DataCommunications
micom MicroComputers for DataCommunications
micom MicroComputers for DataCommunications
micom MicroComputers for DataCommunications
MICRONETIC
A DATA COMMUNICATION EXPERTISE COMPANY
FOR MORE IMFORMATION AND FREE CATALOG
WRITE TO MICRONETIC COMPANY LIMITED
(IMPORT PRODUCTS DIV.)
28/43-45 VIBHAVADI RANGSIT RD.,
BANGKEN BANGKOK 10900
OR : TEL. 5110759 , 5114290
micronetic

BMI Power Line Monitor
เครื่องตรวจปัญหาไฟ AC ระบบคอมพิวเตอร์ที่คุณคุ้มค่าและควรมี



ถ้าคุณมีปัญหาเครื่องมือรวนหรือทำงานผิดพลาดโดยไม่รู้สาเหตุ **BMI Power Line Monitor** ซึ่งเป็นเครื่องมือตรวจสอบปัญหาทางระบบไฟฟ้า จะบอกคุณได้ว่าเครื่องมือของคุณขัดข้องจากปัญหาทางระบบไฟฟ้าหรือปัญหาจากอุปกรณ์ในเครื่อง เพื่อการแก้ไขที่ถูกต้องและรวดเร็ว

ตัวอย่างปัญหาและวิธีการแก้ไขที่ถูกต้องจาก BMI Power Line Monitor

ปัญหา	สาเหตุ	การแก้ปัญหา
ไฟดับ **(Power Failure)**	ไม่ได้เสียบปลั๊กเครื่องหรือปลั๊กหลวม ระบบจ่ายไฟฟ้าล้มเหลว, สวิทช์ตัดต่อถูกตัดวงจร	ใช้ Gould UPS (Uninterruptible Power System) ซึ่งมีแบตเตอรี่จ่ายพลังงานชดเชยให้ แยกสวิทช์ตัดต่อวงจร หรือ ใช้สวิทช์ตัดต่อที่มีกำลังสูงขึ้น
แรงดันไฟตก	ใช้ไฟมากเกินไปจน overload หรือเปิดเครื่องที่มีมอเตอร์ใหญ่กินไฟมาก	ใช้เครื่องปรับแรงดันไฟฟ้า Gould Line Conditioner หรือแยกวงจรตัดต่อ
แรงดันไฟฟ้าสูง	เกิดการผิดพลาดในระบบสายไฟหรือระบบจ่ายไฟ	ใช้เครื่องปรับแรงดันไฟฟ้า Gould Line Conditioner หรือแก้ไขการเดินสายไฟให้ถูกต้อง
Voltage Spike	มีการตัดต่อโหลด ที่เป็น Motor Compressor ขนาดใหญ่ ฟ้าผ่า ฟ้าแลบ หรือไฟฟ้าสถิตย์	ใช้เครื่องตัดสัญญาณรบกวน Gould Super Isolation transformer หรือเครื่องปรับแรงดันไฟฟ้า Gould Line Conditioner ซึ่งใช้เทคนิคหม้อแปลงแบบแยกส่วน
Voltage Drop	มีการเปิดเครื่องที่ใช้มอเตอร์ใหญ่ ๆ มีการตัดต่อในระบบจ่ายไฟฟ้าใหญ่ๆ มีการเพิ่มโหลดใหญ่ๆ หรือเกิดการขัดข้องของระบบจ่ายไฟฟ้า	ใช้เครื่องปรับแรงดันไฟฟ้า (Gould Line Conditioner)

เรายินดีให้คำปรึกษาและสาธิตวิธีการให้ชมได้ตลอดเวลา สนใจติดต่อ

บริษัท แมสเวอลด์ จำกัด
ตู้ ป.ณ. 12-144 กทม. โทร.(02) 2349330

ยูนิเทค ผู้บุกเบิกภาษาไทย ขอเสนอ
ระบบ ภาษาไทย รุ่นใหม่ ขนาด 40 คอลัมน์
(ไทย version 2.2)

ระบบภาษาไทย version 2.2 นี้ เป็นระบบภาษาไทยสำหรับ APPLE COMPATIBLE COMPUTER ซึ่งสามารถใช้กับ program ภาษา APPLE SOFT (ภายใต้ DOS 3.3 operating system) ได้ทันที โดยสามารถแสดงผลเป็นภาษาไทยและภาษาอังกฤษ ได้ 12 บรรทัดๆละ 40 ตัวอักษร และสามารถจัดระดับของตัวอักษรภาษาไทยได้อย่างอัตโนมัติทั้ง 4 ระดับโดยอิสระต่อกัน (ระบบภาษาไทยอื่นๆ มักจะจัดระดับของตัวอักษรภาษาไทยได้เพียง 3 ระดับโดยอิสระต่อกัน) ทั้งบนจอภาพ และ เครื่องพิมพ์

การแสดงผลบนจอภาพอาศัย Bit-mapped character generator subroutine ควบคุมการจัด line ของตัวอักษรภาษาไทย ซึ่ง subroutine นี้เขียนขึ้นโดยภาษา assembly ทั้งหมด จึงไม่มีปัญหาด้านความเร็วในการแสดงตัวอักษรทั้ง 2 ภาษา (การจัด line นี้จะเป็นไปโดยอัตโนมัติ แม้ว่าจะเป็นขณะที่กำลังอยู่ในคำสั่ง INPUT ก็ตาม เพราะ subroutine สำหรับจัด line นี้ถูกควบคุมโดย DOS 3.3 โดยตรง) และสามารถผสมอักษรบน graphic page ได้ทั้ง 2 pages ในเวลาเดียวกันได้ โดยผู้ใช้สามารถเลือก display mode ได้ถึง 10 แบบด้วยกัน

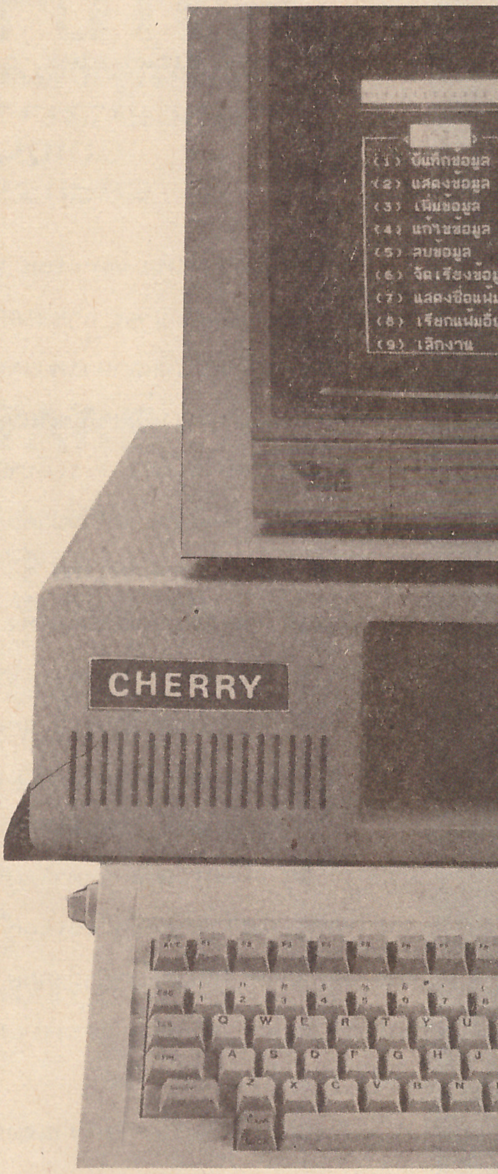
นอกจากนี้ยังมี FUNCTIONS สำหรับควบคุม character generator subroutine ดังกล่าวมากกว่า 20 FUNCTIONS จึงทำให้ผู้ใช้สามารถใช้ภาษาไทยได้ง่ายและสะดวกกว่าระบบภาษาไทยอื่นๆ เช่น สามารถใช้ full screen editor ได้ , สามารถ delete และ insert ข้อความขณะ input line ใดๆ ได้ทั้งภาษาไทยและภาษาอังกฤษ (ระบบภาษาไทยอื่นๆ สามารถ delete ได้แต่ insert ไม่ได้) , สามารถ clear screen และเปลี่ยน display mode or page ได้โดยอาศัย control character เพียง 1-2 ตัวเท่านั้น เป็นต้น

และเนื่องจากการทำงานของระบบภาษาไทย version 2.2 นี้ได้ถูกออกแบบมาเพื่อให้ DOS 3.3 operating system สามารถควบคุมได้โดยตรง ดังนั้นจึงสามารถใช้ชื่อ file เป็นภาษาไทยได้โดยไม่มี error ในการค้นหาชื่อ file

สนใจติดต่อชมการสาธิตได้ที่ บริษัท ยูนิเทค (ประเทศไทย) จำกัด 1087/146 ปากซอย จารุวัฒน์ ถนน เพชรบุรี ตรงข้ามธนาคาร ไทยพาณิชย์ สาขาใหม่ กรุงเทพฯ โทร. 251-6465

*** ระบบภาษาไทย version 2.2 นี้ พัฒนาขึ้นโดย ภาสุทัต เตชะเสน บริษัท ยูนิเทค (ประเทศไทย) จำกัด ***

คุณลักษณะ 9 ประการของ CHERRY III—FX



MULTIPROCESSOR

มีไมโครโปรเซสเซอร์ 4 ตัวในเครื่องเดียวกันหมายความว่าสามารถทำงานได้มากกว่า 2 ระบบ คือ ระบบ CP/M 2.2 CP/M 3.0 และโปรแกรมสำเร็จรูปทุกชิ้นของ APPLE II/II-E

ERGONOMIC KEYBOARD WITH FUNCTION

คีย์บอร์ดแบบ DETACHABLE ขนาด 95 คีย์พร้อม 70 FUNCTION รวมทั้งแป้นภาษาไทยปรับระดับได้ 2 ตำแหน่ง

RELIABILITY

มีระบบป้องกันถึง 3 ชั้นสำหรับคอมพิวเตอร์อันมีค่า คือระบบป้องกันแรงดันไฟเกิน-ไฟกระโชก-ป้องกันสัญญาณรบกวน ช่วยยืดอายุคอมพิวเตอร์ให้ยาวนาน

THAI LANGUAGE WITH WINDOW DISPLAY

เป็นเครื่องแรกที่มีระบบภาษาไทยในตัวถึง 2 ระบบ เลือกใช้ได้คือ ระบบภาษาไทย CP/M 80 ตัวอักษร 8 บรรทัด และระบบภาษาไทย APPLESOFT-DOS 40 ตัวอักษร 12 บรรทัด สามารถปนภาษาอังกฤษและภาพกราฟฟิคและได้คิดค้นโดยผู้เชี่ยวชาญจากยูนิเทค

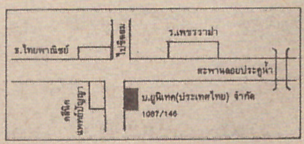
MULTIPLE TERMINAL

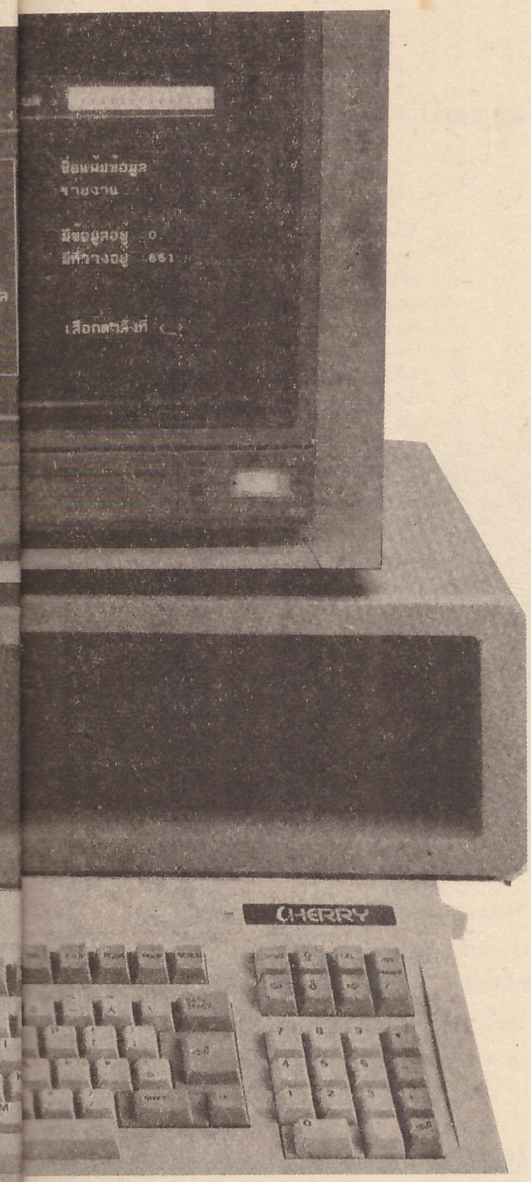
ด้วย HARD DISK ระบบ UNIBUS TM ทำให้สามารถต่อ CHERRY-EX ได้อีก 40 STATION หรือ MULTI-USER, MULTI-OPERATING SYSTEM

บริษัท ยูนิเทค (ประเทศไทย) จำกัด

1087/146 ปากซอยจารุรัตน์ ถ.เพชรบุรี ตรงข้ามธนาคารไทยพาณิชย์

สาขาใหญ่ กรุงเทพฯ โทร. 251-6465





DYNAMIC MEMORY

หน่วยความจำมาตรฐานขนาด 256K ไบท์สามารถได้ถึง 384 Kไบท์ ในส่วนของ 256 Kไบท์ นี้ท่านสามารถนำมาใช้เป็น SOLID-STATE RAM DISK หรือดิสค์ไดรฟ์อีกตัวหนึ่งที่ให้ความเร็วในการทำงานสูงกว่าไดรฟ์ธรรมดา 10 เท่า!และปราศจากข้อผิดพลาด เหมาะสำหรับโปรแกรมที่ใช้การค้นหาข้อมูลมากและรวดเร็ว

SOFTWARE SUPPORT

มีโปรแกรมสำเร็จรูปปัจจุบันถึง 12,000 ชิ้น (ภาษาอังกฤษ) และโปรแกรมภาษาไทยหลายรายการในระบบ 40 คอลัมน์ รายละเอียดดูรายการโปรแกรมภาษาไทย

MASS STORAGE DISK DRIVE

เป็นดิสค์ไดรฟ์แบบ DOUBLE SIDE ดูความจุ 327K ต่อ 1 ตัวหรือ 654K ต่อคู่หรือดิสค์ไดรฟ์ รุ่นธรรมดาความจุ 160K ต่อตัว 320K ต่อคู่ (OPTIONAL)

EXPANDIBILITY

• MULTI-BOARD TM —— จาก DTC สามารถใช้ระบบเพิ่ม 16 บิทให้กับเครื่องด้วย MS-DOS CP/M-86 P-UCSD SYSTEM หรือ PC-DOS และโปรแกรมทุกชิ้นของเครื่องIBM-PCได้

รายการโปรแกรมภาษาไทย 40 คอลัมน์ และ 80 คอลัมน์

โปรแกรมเก็บทะเบียนและประวัตินักศึกษา (40)
โปรแกรมบริหารโรงพยาบาล (40)
โปรแกรมบริหารร้านวีดีโอ (40)
โปรแกรมบัญชีรับ-จ่าย (80)
โปรแกรมบัญชี ลูกหนี้-เจ้าหนี้ (80)
โปรแกรมระบบควบคุมสต๊อค (80)
โปรแกรมระบบบริหารธุรกิจสำนักงาน (40)
โปรแกรม DBASE II ภาษาไทย (80)
โปรแกรมจัดการฐานข้อมูล (40)
ฯลฯ

PC-5500 11,500.-

- 64 K PERSONAL COMPUTER
- 100% EXPANDABLE UPTO 256 K RAM
- 10 USER DEFINE KEY (BATTERY BACK UP)
- 188 FUNCTION KEY :
  - 94 BASIC FUNCTIONS
  - 94 CP/M FUNCTIONS
- AUTO REPEAT KEY ทุก KEY
- AUTO LINE NUMBER (10, 20, 30)
- FUNCTION CONTROL KEY
- CAGE พลาสติกล้วน สีนุ่มนวลและเป็น FURNITURE ได้ในตัว

PC-5500

ของเรามี 2 แบบให้ท่านเลือก ทุกรุ่นใช้ Keyboard แบบเดียวกัน แตกต่างกันที่ Main Board ตามความเหมาะสมกับงานของท่าน

PC-5500-1

ซึ่งใช้ Main Board ของ AP II ธรรมดา ซึ่งมีคุณสมบัติเทียบเท่า APPLE และเล่นโปรแกรมของ APPLE II และ CP/M ได้ทุกโปรแกรม

11,500.-

PC-5500-2

- เป็นแบบ 64 K RAM Built-in, Single CPU
- มี SLOT อยู่ครบ 8 SLOTS
- ใช้โปรแกรม Apple II และ CP/M ได้ทุกโปรแกรม

12,300.-

PC-5500

ทุกรุ่นสามารถใส่ Disk Drive เข้าไว้ในเครื่องได้ 2 เครื่อง มีแป้นสำหรับติดพัดลมระบายอากาศอยู่ในกล่องเรียบร้อย

PC-5500

ทุกรุ่น สามารถขยายหน่วยความจำในการ RUN โปรแกรม VISICALC จาก 34K เป็น 145K เมื่อใช้แผงวงจร 128K RAM card 1 แผงและสามารถ SAVE FILE ที่มีความยาวมาก ๆ ลงแผ่น DISKETTE ได้มากกว่า 1 แผ่น



PC-5500 BUSINESS COMPUTER

:: ใน MODE CP/M ทั้งหมด สามารถใช้ 128 K RAM Card ได้ 2 แผง ::

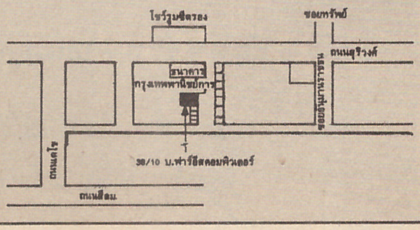
- ใช้งานเป็น DISK DRIVE เทียม เรียกว่า "PSEUDO DISK" ที่สามารถ RUN โปรแกรม ได้เร็วกว่าธรรมดาหลาย 10 เท่า
- เหมาะสำหรับโปรแกรม WORDSTAR, dBASE II MULTIPLAN
- การค้นหา DATA (SEARCH) ใน FILE ที่มีความยาวมาก ๆ จะประหยัดเวลาได้มากทีเดียว ทำให้ตัดปัญหาในเรื่องเวลาให้หมดไปโดยสิ้นเชิง
- เหมาะกับ Application ที่ใช้งานแบบ ON-LINE ซึ่งค้นหาข้อมูลและบันทึกข้อมูลได้เร็ว

ท่านที่สนใจการใช้งาน AP II ที่สามารถขยาย MEMORY ได้ถึง 256 K

โปรดสอบถามรายละเอียดที่บริษัทได้ตลอดเวลา

บริษัท ฟาร์อีสท์ คอมพิวเตอร์ เอ็นยีเนียริ่ง จำกัด

38/10 ซอยอนุมานราชธน
ถนน สุริวงศ์ บางรัก กทม. 10500
โทร. 235-9230-1



มี COURSE ภาษา BASIC สำหรับเด็ก

COM-๑๘

โรงเรียนคอมพิวเตอร์ศึกษา

(คอมเอ็ด) ชั้น 6 อาคารปริญญา 544/11 ถนนเพลินจิต กรุงเทพฯ 10500
โทร. สายตรง 2515503 โทร. 2515505-6 ต่อ 65

วงการธุรกิจต่าง ๆ ได้นำเอาคอมพิวเตอร์มาใช้อย่างแพร่หลายจึงเกิดความต้องการบุคลากรทางด้านคอมพิวเตอร์ เพื่อความก้าวหน้าในอนาคต ตัดสินใจเรียนเสียแต่บัดนี้



ทำไมเขาถึงเลือกเรียนที่คอมเอ็ด

เราเปิดมากว่า 2 ปี มีประสบการณ์เป็นอย่างดี
เน้นภาคปฏิบัติ 1 คน ต่อ 1 เครื่อง
เพื่อให้ได้ผลเรียนกลุ่มละไม่เกิน 15 คน
จบแล้วมีประกาศนียบัตรรับรองโดยกระทรวงศึกษาธิการ
แจกตำราและอุปกรณ์การเรียนฟรี
ฝึกใช้คอมพิวเตอร์นอกเวลาเรียนได้ฟรี
ชดเชยชั่วโมงเรียนถ้าตรงกับวันหยุด
สอนตั้งแต่ขั้นพื้นฐานจนประกอบอาชีพได้
ติดเครื่องปรับอากาศ บรรยากาศน่าเรียนและเป็นกันเอง
หลักสูตรตรงตามความต้องการและใช้งานได้จริง

- ใช้คอมพิวเตอร์รุ่นใหม่แทนบัตรเจาะ
- สอนโดยอาจารย์ผู้ทรงคุณวุฒิทางคอมพิวเตอร์
- ไปมาสะดวก มีรถเมล์ผ่านหลายสาย มีที่จอดรถกว้างขวาง
- เรียนครบ 3 คอร์ส แถมเรียน SUPERCALC ฟรี

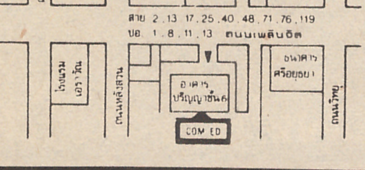
วิชาที่เปิดสอน

1. INTRODUCTION TO COMPUTER
2. PROGRAMMING IN BASIC
3. CP/M
4. SUPERCALC
5. dBASE II (มีภาษาไทยและอังกฤษ)
6. WORDSTAR
7. BUSINESS BASIC (ไทยและอังกฤษ)
8. COBOL
9. ADVANCE COBOL
10. FORTRAN
11. การวางระบบโปรแกรมบัญชีแยกประเภท
12. การวางระบบโปรแกรมจ่ายเงินเดือน
13. ซ่อม สร้าง ประกอบคอมพิวเตอร์

รอบเช้า 10.00-12.00 น.
รอบบ่าย 13.00-16.00 น.
รอบค่ำ 18.00-20.30 น.
มีกลุ่ม จันทร์-พุธ-ศุกร์
และ อังคาร-พฤหัส

พิเศษ! มี COURSE ภาษา BASIC สำหรับเด็ก

สมัครเรียนได้ตั้งแต่ จันทร์-เสาร์ 10.00-18.00 น.



สถาบันคอมพิวเตอร์ คอม-เอ็ด

COM-ED INSTITUTE OF COMPUTER

หลักสูตรโดยได้รับอนุมัติจากกระทรวงศึกษาธิการ
สำเร็จหลักสูตรแล้วได้รับ ประกาศนียบัตรรับรองโดยกระทรวงศึกษาธิการ

DISAK COMPUTER

ศูนย์หนังสือ ไมโครคอมพิวเตอร์ AP-II

จำหน่ายปลีกและส่ง รับส่งทางไปรษณีย์

ต่างจังหวัดสั่งจ่ายธนาณัติในนาม

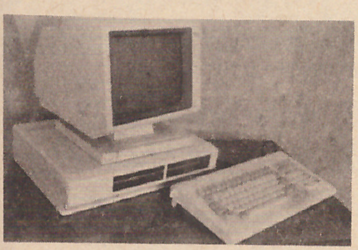
นาย ดิศักดิ์ ศรีสุวรรณ ปณ.ตลาดพลู

ค่าส่งเล่มละ 5 บาท

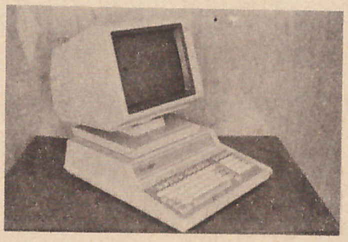
	ITEM	PRICES		ITEM	PRICES		ITEM	PRICES
001	APPLESOFT TUTORIAL MANUAL	70	101	APPLE PILOT (LANGUAGE REF. MANUAL)	120	162	CP/M-86 COMMAND SUMMARY	50
002	APPLESOFT REFERENCE MANUAL	80	102	32 BASIC PROGRAMS FOR APPLE	150	163	WORLD HANDER	30
003	APPLE REFERENCE MANUAL	80	103	THE DATA REPORTER	140	164	APPLE WORD	40
004	APPLE DISK OPERATION MANUAL	80	104	SUPER-TEXT II	110	165	BASIC MAILER	65
005	APPLE FORTRAN	90	105	SUPER-SCRIBE USER'S MANUAL	120	166	INVENTORY CONTROL SYSTEM	100
006	APPLE PASCAL	90	106	APPLE II SERIAL INTERFACE CARD	30	167	16K RAM CARD	100
007	MICROSOFT Z-80 VOL.1	90	107	APPLE WRITER	60	168	APPLE-CILLIN II	40
008	MICROSOFT Z-80 VOL.2	90	108	APPLE HOW TO!	60	169	QUICKCODE DBASE II	100
009	APPLE II PROGRAMMER HANDBOOK	90	109	Z-80 AND 8080 ASSEMBLY LANGUAGE	140	170	THE DBASE WINDOW	120
010	APPLE II USER'S GUIDE	120	110	STARTING FORTH	140	171	ABSTAT	100
011	USING 6502 ASSEMBLY LANGUAGE	180	111	MICROSOFT MULTIPLAN	240	301	GUIDE TO OPERATION	250
012	APPLE MECHANIC	60	115	LISA 2.5 FOR APPLE II & II PLUS	70	302	BASIC (FOR IBM PC)	320
016	THE HOME ACCOUNTANT	70	116	CP/M BASIC COMPILER	70	303	DISK OPERATION SYSTEM	300
022	CP/M COBOL-80 FOR APPLE II	160	117	APPLE MUSIC THEORY	90	304	LOTUS 123 CUSTOMER ASSURANCE PLAN	420
023	CP/M FORTRAN-80 FOR APPLE II	140	118	VISI-SCHEDULE	150	305	YOUR IBM PC	300
024	DBASE MASTER MANUAL (+UTILITY)	100	119	HOW TO REPAIR THE APPLE COMPUTER	50	306	IBM PC EXPANSION & SOFTWARE GUIDE	360
025	VISICALC MANUAL	90	120	DOING BUSINESS WITH VISICALC	160	307	PERFECT FILER	180
026	VIDEX 80-COLUMN MANUAL	70	121	DBASE II USER'S GUIDE	160	312	IBM TECHNICAL REF. MANUAL	320
027	APPLE BASIC FOR BUSINESS	130	122	SCREEN WRITER II	120	139	DOCUMENT/PLUS (FOR WORDSTAR)	90
028	WORDSTAR	310	123	THE DBASE-II PROGRAMMER'S UTILITY	140	140	CP/M PRIMER	100
029	APPLE PASCAL (OPERATION SYS.)	110	124	REPLAY	50	141	THE WILDCARD	40
030	APPLE II MONITOR PEELED	70	125	KIDS AND THE APPLE	160	142	KRELL'S M.I.T LOGO (FOR APPLE II)	60
031	APPLESOFT TOOL KIT	30	126	VISIFILE	90	143	THE APPLE II CIRCUIT DESCRIPTION	240
032	6502 ASSEMBLER EDITOR	40	127	MT+ CP/M PASCAL	140	144	THE POWER OF MULTIPLAN	180
033	APPLE PLOT	40	128	TASC APPLESOFT BASIC COMPILER	80	145	GRAPHICS COOK BOOK FOR THE APPLE	120
037	NIPPLE EXPRESS VOL.1-2	240	129	APPLE PIE	180	146	FORMAT II	180
038	WORDSTAR MADE EASY	80	130	SUPER PLOTTER & SCIENTEFIC PLOTTER	100	147	CP/M REVEALED	160
039	PFS- USER'S MANUAL & REPORT	80	131	CP/M SIMPLIFIED	140	148	DOING BUSINESS WITH SUPERCALC	160
040	BENEATH APPLE DOS	100	132	INSIDE CP/M	240	149	CP/M HANDBOOK WITH MP/M	200
041	CCA DATA MANAGEMENT SYSTEM	100	133	HOW TO USE THE SUPERCALC	120	150	CP/M USER GUIDE (FOR ALL CP/M USERS)	200
043	VISITREAN+VISIPLOT	100	134	CP/M BASIC COMPILER UTILITY	90	151	APPLE MACHINE LANGUAGE	200
044	APPLE LANGUAGE SYSTEM	40	135	55 CP/M UTILITIES IN ONE PACKAGE	120	152	EASYFROM-A BUSINESS FROM PROCESSOR	30
045	APPLE LOGO (REF.MANUAL)	90	136	INTRODUCTION TO WORDSTAR	180	153	DOS BOSS COMMAND EDITOR	30
046	APPLE LOGO (INSTRUCTIONS)	80	137	EVERYMAN'S DATABASE PRIMER	200	154	ACCOUNT RECEIVABLE SYSTEM FOR AP	100
047	BAG OF TRICKS	70	*	(FEATURING DBASE II)	-	155	ALPHA PLOT (HI-RES APPLE GRAPHIC)	30
048	APPLE II ASSEMBLY LANGUAGE	140	138	COPY PROGRAM PERFECT	100	156	AP FILE & APPLE TEST	30
050	WHAT'S WHERE IN THE APPLE?	100	066	MIND MEMORY IMPROVEMENT CORSE	50	157	PROGRAM LINE EDITOR	30
054	WORDSTAR CUSTOMIZATION NOTES	120	067	3-D SUPERGRAPHICS & GAME TOOL	100	158	THE HOME MONEY MINDER	30
055	SUPER-SORT 1.5	120	068	APPLE POST	40	172	64K/128K RAM BOARD O/M	100
056	CALCSTAR	120	069	APPLE PILOT (EDITOR MANUAL)	120	173	PERSONAL FINANCE MANAGER	40
057	DATASTAR	170	070	THE BOOK OF APPLE SOFTWARE 1983	240	174	ANIMATION PACKAGE	60
058	EASY WRITTER & LETTER WRITTER	80	071	LOGO (FOR THE APPLE II)	140	175	VISICALC BUSINESS FORECASTING MODEL	100
059	THE CASHIER	90	072	APPLE WRITTER II	70	176	MAGIC WINDOW	80
060	THE CONTROLLER	180	073	ALD SYSTEM II	40	177	PL/I-80 APPLICATION GUIDE	360
061	THE POWER OF SUPERCALC	100	074	THE POWER OF VISIPLOT	130	178	STRIPCHARTER	100
062	DBASE II (USER'S MANUAL)	90	075	VISICALC (HOME & OFFICE COMPANION)	150	179	FORTH-79 VERSION 2	80
063	DBASE II (REF.MANUAL)	90	076	THE VISICALC BOOK	150	180	NIBBLE EXPRESS III	200
065	THE POWER OF VISICALC I & II	140	077	PFS- FILE (UPDATE VERSION)	70	181	MICROSOFT MS-DOS OPERATING SYSTEM	380
089	VISIACCOUNT	150	078	ASSEMBLY LANGUAGE PROGRAMMING	90	182	dGRAPH	200
090	VISILEDGER	150	081	PEACHTREE (GENARAL LEDGER SYS.)	150	183	POLISHING YOUR APPLE	120
091	ANALYZER	150	082	PEACHTREE (PAYROLL SYS.)	150	185	STOCKSELLER OPERATOR'S GUIDE	160
092	DATA PERFECT	150	083	PEACHTREE (ACCOUNT RECEIVABLE SYS.)	150	186	STOCKFILE OPERATOR'S GUIDE	200
093	VISITERM	100	084	PEACHTREE (INVENTORY SYS.)	150	313	UCSD PASCAL REFERENCE	300
094	DESTOP PLAN	70	085	PEACHTREE (PAYABLE SYS.)	150	314	FORTRAN-77 REFERENCE	300
095	VISIDEX	80	087	THE ORDER SCHEDULE	30	315	UPDATES FOR IBM PC COBOL COMPILER	320
096	MICROBOOK- DATA MANAGEMENT FOR APPLE	200	088	FORTH 79	80	316	BEGINNER'S GUIDE	250
097	APPLE GRAPHICS & ARCADE GAME DESIGN	170	112	APPLE II/III SOOOFTWARE DIRECTORY	380	317	ASSEMBLER REFERENCE	200
098	GRAPHIC SOFTWARE FOR MICROCOMPUTER	180	159	CP/M-86 SYSTEM GUIDE	200	318	FORTRAN COMPILER	220
099	PFS- GRAPH	70	160	CP/M-86 USER'S GUIDE	200	319	SUPERCALC 2 USER'S GUIDE & REF. MANU	200
100	WORDSTAR TRAINING GUIDE	100	161	CP/M-86 PROGRAMMER'S GUIDE	200	320	HARDWARE MAINTENANCE AND SERVICE	400

COMP TOWN Co., Ltd.
บริษัท คอมพ์ ทาวน์ จำกัด

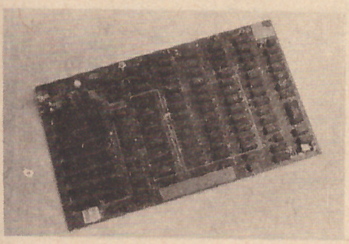
จำหน่ายและบริการให้คำปรึกษาเกี่ยวกับเครื่องคอมพิวเตอร์ โดยเฉพาะเครื่องในระดับไมโครคอมพิวเตอร์ ทั้งทางด้าน **HARDWARE และ SOFTWARE**



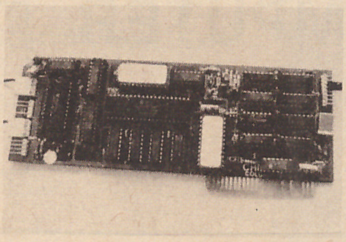
AP-II JAPAN Case PC-5500 + Key Board P.C. + Monitor New model



AP-II JAPAN RX-8800 188 Function Key + Monitor



MAIN BOARD JAPAN AP-II



80-Column + AUTOBOOT without Softswitch

```
************************
*  COMP TOWN Co.,Ltd.  *
************************
```

TEL:279-3808

ITEM	DESCRIPTION	PRINCE/UNIT
1.	AP-II 48KRAM 52KEY JAPAN (NEWTECH)	12,000 BAHT
2.	AP-II 48KRAM RX-8800 JAPAN (188 FUNCTION KEY)	13,500 BAHT
3.	AP-II 48KRAM KEYBOARD P.C. JAPAN	14,500 BAHT
4.	DISK-DRIVE EPSON AP-200 WITH CONTROL/CARD	20,000 BAHT
5.	DISK-DRIVE SUPER-5 WITH CONTROL/CARD	10,000 BAHT
6.	DISK-DRIVE NATIONAL WITH CONTROL/CARD	10,000 BAHT
7.	GREEN MONITOR 12" (YJE)	3,800 BAHT
8.	GREEN MONITOR 12" (YJE) NEW MODEL	4,500 BAHT
9.	GREEN MONITOR 12" DISCO	3,800 BAHT
10.	16KRAM CARD (JAPAN)	1,500 BAHT
11.	128KRAM CARD WITH 5 PROGRAM	5,200 BAHT
12.	Z-80 CARD (JAPAN)	1,500 BAHT
13.	80-COLUMN + AUTOBOOT	2,500 BAHT
14.	PRINTER EPSON FX-80 WITH CARD/C	19,000 BAHT
15.	PRINTER EPSON FX-100 WITH CARD/C	29,000 BAHT

9/1-2 ข้าง อ.ต.ก.ตรงข้างสวนจตุจักร ถนนพหลโยธิน สะพานควาย กรุงเทพ 10400 โทร. 279-3808

CUB

สถาบันอบรม "คอมพิวเตอร์" ธุรกิจ
COMPUTER USE for BUSINESS TRAINING INSTITUE

โทร. 279-3808 9/1-2 ข้าง อ.ต.ก. ตรงข้ามสวนจตุจักร ถ.พหลโยธิน กรุงเทพ 10400

เปิดอบรมวิชาคอมพิวเตอร์ วิชาละ 32 ช.ม.

ใช้เวลาฝึกอบรมประมาณ 8 สัปดาห์ ตามรายละเอียดดังนี้

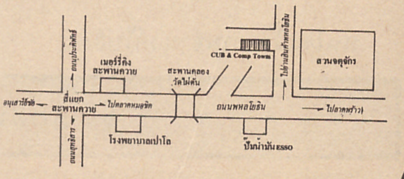
(ยกเว้นวิชาที่มีเครื่องหมาย ใช้เวลา 18 ช.ม.)*

วิชา

1. Introduction to Computer
2. BASIC Programming
3. Advanced BASIC Programming
4. COBOL Programming
5. Business System Analysis
6. MIS for General Manager
7. *Computer Assisted Secretary
8. *CP/M
9. *Multiplan
10. *dBASE II
11. Accounting for Computer Application (หลักการบัญชีสำหรับงานคอมพิวเตอร์)
12. Micro Computer Hardware

**(รวมตำราและอาหารว่างทุกวัน)

(สำหรับวิชา Computer for Kids สถาบันจะเปิดครั้งต่อไปในเดือนมีนาคม 2527)



ปิดเทอมนี้
เรียนคอมพิวเตอร์ที่
BCC คุ้มค่ากว่า

BCC
The Business Computer Centre

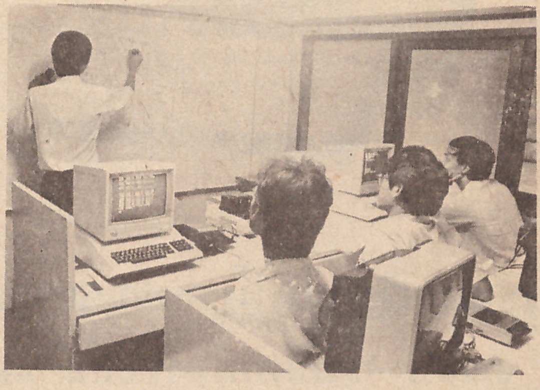
ศูนย์อบรมคอมพิวเตอร์ธุรกิจ

ในความควบคุมของกระทรวงศึกษาธิการ

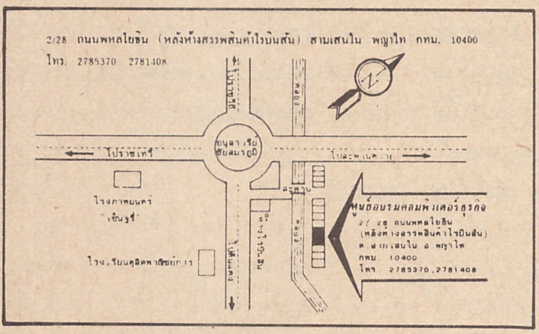
โทร.278-5370, 278-1408

ในช่วงภาคฤดูร้อน หรือช่วงปิดเทอมนี้ ทาง BCC ได้จัดอบรมคอมพิวเตอร์คอร์สพิเศษขึ้น โดยมีวิชาดังต่อไปนี้

1. **INTRODUCTION TO COMPUTER**
2. **KEY OPERATOR**
3. **BASIC PROGRAMMING**
4. **ADVANCED BASIC PROGRAMMING**
5. **PACKAGE** (โปรแกรมสำเร็จรูป)
- **CP/M SUPERCALC, D-BASE II** และ **WORDSTAR**



- ทุกวิชาเน้นภาคปฏิบัติกับเครื่อง AP II และ COMMODORE พร้อมทั้งแจกตำราเรียนมูลค่ากว่า 200 บาท
- นอกจากนี้ เรายังมีเกมส์ไว้ให้ท่านพักผ่อนนอกเวลาเรียน หรือจะใช้เครื่องทำงานที่คั่งค้างของท่านก็ยังได้
- ทุกวิชามีการสอบ เพื่อรับใบประกาศนียบัตร ซึ่งรับรองโดยกระทรวงศึกษาธิการ จึงเป็นหลักประกันได้ว่าทุกท่านที่สำเร็จออกไปจะต้องมีความรู้จริง



เวลาเรียน

รอบ
- จันทร์-ศุกร์ 10.30-12.00 น.
- จันทร์-ศุกร์ 12.00-13.30 น.
- จันทร์-ศุกร์ 13.30-15.00 น.
- จันทร์-ศุกร์ 17.30-19.00 น.
- จันทร์-ศุกร์ 19.00-20.30 น.
- เสาร์ 13.00-15.30 + อาทิตย์ 9.30-12.00 น.
- เสาร์ 15.30-18.00 + อาทิตย์ 13.00-15.30 น.

รีบสมัครด่วน! รับเพียงกลุ่มละ 14 คน เท่านั้น

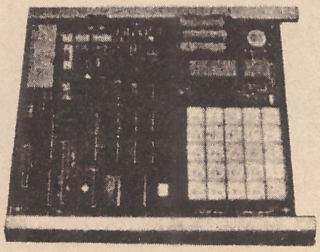
จองที่เรียนโทร.278-5370, 278-1408 หรือที่
2/28 หลังโรบินสัน อนุสาวรีย์ชัยฯ ถนนพหลโยธิน เขตพญาไท กทม.10400

พิเศษ! บริการจัดหาเครื่องและรับเขียนโปรแกรมในราคาพิเศษ

มาสร้าง ไมโครคอมพิวเตอร์ ด้วยตัวคุณใช้เองดีกว่า ได้ทั้งความรู้ ความเพลิดเพลิน และความภูมิใจที่สร้างขึ้นมาด้วยตัวของเรา

"Perfect Z80" SINGLE BOARD MICROCOMPUTER

เป็น board ที่สมบูรณ์แบบที่สุดทั้งทางด้าน Hardware และ Software



คุณสมบัติของ Perfect Z 80

1) ใช้ Z 80 A เป็น CPU
2) Memory มีขนาดสูงถึง 20KB แบ่งเป็น ROM12KB, RAM 8KB- (ทำให้มีหน่วยความจำเพียงพอในการใช้ภาษา BASIC ออกจอ TV ได้อย่างสะดวกมาก)
3) สามารถอัดข้อมูลโปรแกรมลงใน EPROM ได้ทั้งเบอร์ 2716 (2K×8) และ 2732 (4K×8) หรือ Copy
4) ข้อมูลใน RAM สามารถ Back up ได้กรณีปิดไฟดับกระทันหัน โดยใช้ Ni-Cad และมีวงจร charge แบตเตอรี่ได้เองโดยอัตโนมัติ
5) สามารถ Relocate ข้อมูลใน RAM ไปไว้ที่ address ใด ๆ ก็ได้ซึ่งให้ประโยชน์อย่างมากแก่ผู้ใช้
6) Display ให้ LED 7 Segment 6 หลักตัวใหญ่
7) ใช้ Z80A PIO เป็น USER PORT, Z80A CTC เป็นวงจรนับและตั้งเวลาซึ่งโปรแกรมได้
8) มี function มากมายของเครื่องที่พิเศษกว่าเครื่องทั่ว ๆ ไป เช่น Insert, Delete คำนวณ Relative address

ราคาชุด Kit ตั้งแต่ 2,900 บาท ขึ้นไปชุดประกอบสำเร็จ ตั้งแต่ 3,350 บาทขึ้นไป

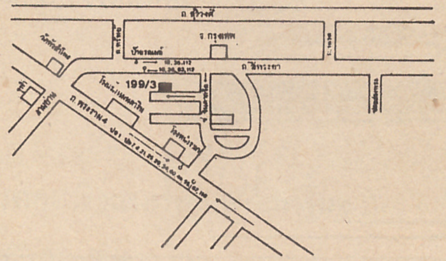
มีอะไหล่สำหรับ นักเล่นอิเลคทรอนิคส์.ไมโครคอมพิวเตอร์ มากมายไว้บริการท่านในราคาพิเศษสุด

- CPU Z 80 A, 6502 และอื่น ๆ
- EPROM ทุกเบอร์ 2716, 2732, 2764, 27128
- RAM ทุกเบอร์ 2114, 6116, 4116
- TTL ทุกเบอร์

โอกาสทองของคุณมาถึงแล้ว

●หากคุณกำลังเบื่อกับ AP II รุ่นเก่า ทั้งในด้านรูปทรงและขีดความสามารถในการทำงาน ขณะนี้เราเปิดโอกาสให้คุณแล้ว บริการรับเปลี่ยนเครื่อง AP II ทุกรุ่น (ไม่ว่าจะเป็นรุ่นไหนก็ตาม) เป็นรุ่นใหม่ล่าสุดของ AP II ที่มีขีดความสามารถสูงสุด คือ รุ่น PC-5500 ในราคาพิเศษสุด โดยนำเครื่องของคุณมาที่บริษัทฯ เพื่อประเมินราคา

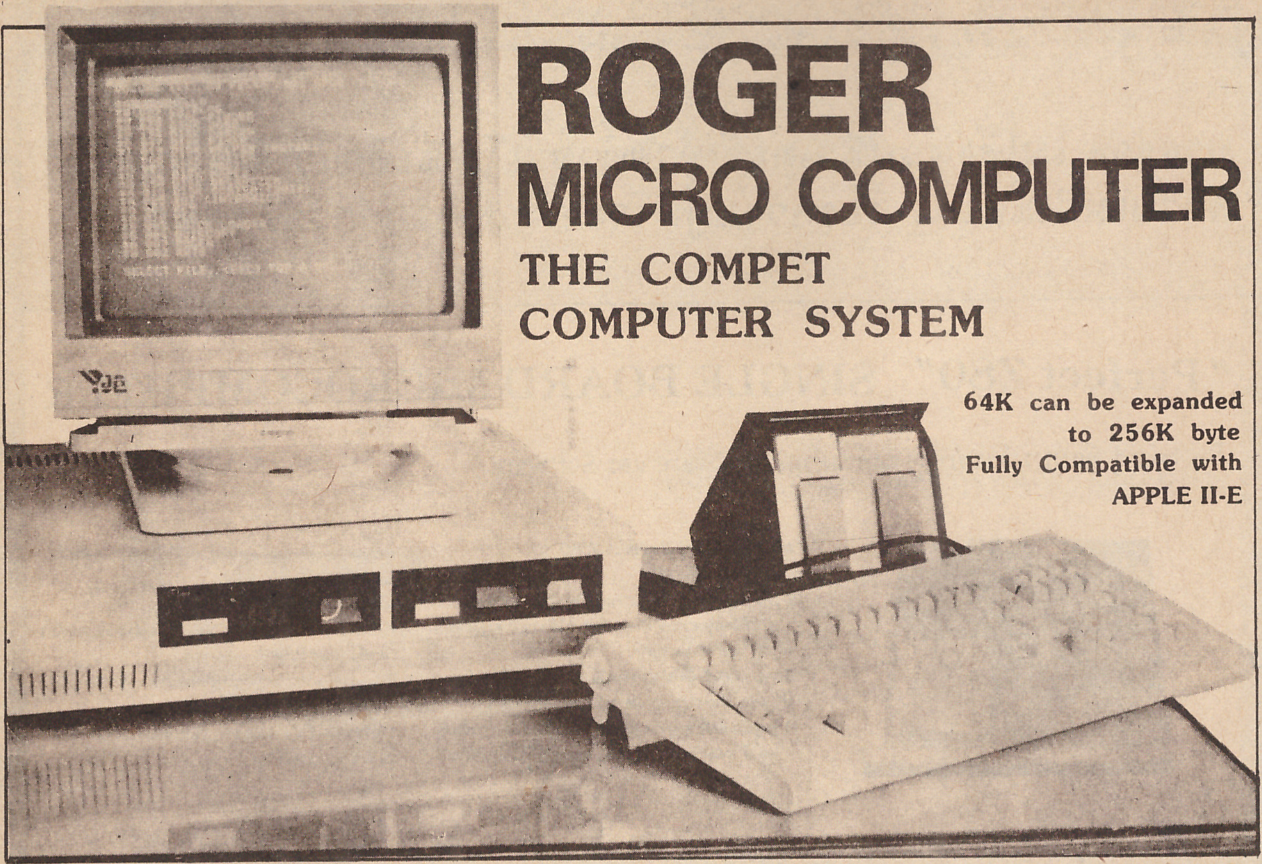
●สำหรับผู้ที่ใช้ AP II โดยใช้ PAL CARD อยู่นั้น ขณะนี้เรากำลังลดราคา Green Monitor เพื่อให้คุณได้ทดลองใช้ในราคาที่ไม่มีผู้ใดทำมาก่อน ด้วยราคาเพียง 3,200 บาท รุ่นยอดนิยม YJE 12" เพื่อให้คุณได้ก้าวขึ้นสู่การใช้งานที่สบายตา ภาพคมชัด



ยินดีรับตัวแทนจำหน่าย

micro HOUSE

ไมโครเฮ้าส์



ROGER
MICRO COMPUTER
THE COMPET COMPUTER SYSTEM

64K can be expanded to 256K byte Fully Compatible with APPLE II-E

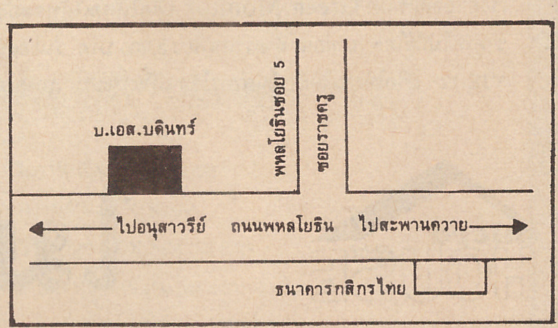


● ข่าวดี 5,500 บาท ●

สำหรับท่านที่ใช้เครื่องของ **APPLE** ต้องการเปลี่ยน **CASE** รุ่นใหม่ แบบ **KEY BOARD** แยก,ในวงเงิน 5,500 บาทเท่านั้น ราคารวม **KEY BOARD** ติดต่อด่วน จำนวนจำกัด

PERIPHERAL

- Z 80
- 80 COLUMN
- 16K RAM CARD
- PAL CARD WITH SOUND
- WILD CARD
- EPROM WRITER CARD
- FUNCTION KEY BOARD
- PRINTER INTERFACE
- DISK DRIVE
- PRINTER EPSON



บริษัท เอส.บดินทร์ จำกัด
1093/1 ใกล้ซอยราชครู ถนนพหลโยธิน กรุงเทพฯ 10400
โทร. 2781589 2785790 2785791

ใบสมัครสมาชิก/ต่ออายุสมาชิก

ชื่อ..นามสกุล..

ที่อยู่..รหัสไปรษณีย์.........................

มีความประสงค์ ☐ สมัครสมาชิก ☐ ต่ออายุสมาชิกหมายเลข.......................

วารสาร ☐ **โนวา** ☐ **เปิดโลก**

สมาชิก กระดาษธรรมดา ☐ 6 เดือน 80 บาท ☐ 1 ปี 150 บาท ☐ 2 ปี 280 บาท

กระดาษปอนด์ ☐ 6 เดือน 90 บาท ☐ 1 ปี 180 บาท ☐ 2 ปี 360 บาท

วารสารไมโครอีเลคทรอนิคส์

สมาชิก กระดาษธรรมดา ☐ 6 เดือน 100 บาท ☐ 1 ปี 200 บาท ☐ 2 ปี 380 บาท

กระดาษปอนด์ ☐ 6 เดือน 120 บาท ☐ 1 ปี 240 บาท ☐ 2 ปี 460 บาท

เริ่มรับตั้งแต่ฉบับที่....................

ส่งธนาณัติ หรือเช็คไปรษณีย์สั่งจ่าย ปท. พระโขนง หรือเช็คธนาคารขีดคร่อมเข้าบัญชีในนามของ "บริษัท สยามซายน์แอนด์เทคโนโลยี่ จำกัด" 1091/230 ศูนย์การค้านครหลวง เพชรบุรีตัดใหม่ กรุงเทพฯ 10400 โทร : 252-2070

MICRO ELECTRONICS

เจ้าของ
บริษัทสยามขายน์แอนด์เทคโนโลยี่ จำกัด
สำนักงาน
เลขที่ 1108/30 ศูนย์การค้าพระโขนง กทม.10110
โทร. 392-2783
ผู้อำนวยการ
สมเกียรติ เจิ่งประภากร
บรรณาธิการ
เสริมพร พูลทรัพย์
ที่ปรึกษา
ยรรยง เต็งอำนวย, ไตรรัตน์ ใจสำราญ
ปริทรรศน์ พันธ์บรรยงก์
กองบรรณาธิการ
จิระพล ฉายัษฐิต สิริชัย ฟูตระกูล
สุวัฒน์ ปุณณชัย อดิสร กรุงเกษม
วศิน เพิ่มทรัพย์
หัวหน้าฝ่ายศิลป์
สุนทร สังข์ทอง
ฝ่ายศิลป์
ปรียา จีนเฉียวฉุน คงศักดิ์ เอมแย้ม
ดุษฎี โชติพฤกษ์
ฝ่ายธุรการ
สุปราณี เดชปัญญาสาร สุมาลี ธรรมเจริญ
ศรัณยา ศิริรัตน์
ฝ่ายโฆษณา
วาสนา ช่วยแก้ว ลาวัลย์ แจ้งอัมพร
สุรพนธ์ ทัพใหญ่
ที่ปรึกษากฎหมาย
เพชราภรณ์ เอื้อวิทยาศุภร
แยกสี
มุมศิลป์
บุญสินธุ์
บริษัท สารมวลชน จำกัด
จัดจำหน่าย
เพ็ญบุญ โทร.278-1317

วารสาร **ไมโครอิเลคทรอนิคส์** เป็นวารสารรายเดือน ● จัดทำโดย บริษัท สยามซายน์แอนด์เทคโนโลยี่ จำกัด ● ราคาจำหน่ายปลีกฉบับละ 20 บาท ● สมัครเป็นสมาชิกได้ในอัตรา : **กระดาษธรรมดา 6 เดือน 100 บาท 1 ปี 200 บาท 2 ปี 430 บาท**
กระดาษปอนด์ 6 เดือน 120 บาท 1 ปี 240 บาท 2 ปี 460 บาท
โดยชำระเงินทางธนาณัติหรือเช็คไปรษณีย์สั่งจ่าย ปท. พระโขนง หรือเช็คธนาคาร ขีดคร่อมเข้าบัญชีในนามของ "**บริษัท สยามขายน์แอนด์เทคโนโลยี่ จำกัด**" และระบุฉบับที่จะเริ่มรับส่งมายัง **บริษัท สยามขายน์แอนด์เทคโนโลยี่ จำกัด** 1091/230 ศูนย์การค้านครหลวง เพชรบุรีตัดใหม่ กรุงเทพฯ 10400 โทร. 252-2070

บ.ก.แถลง

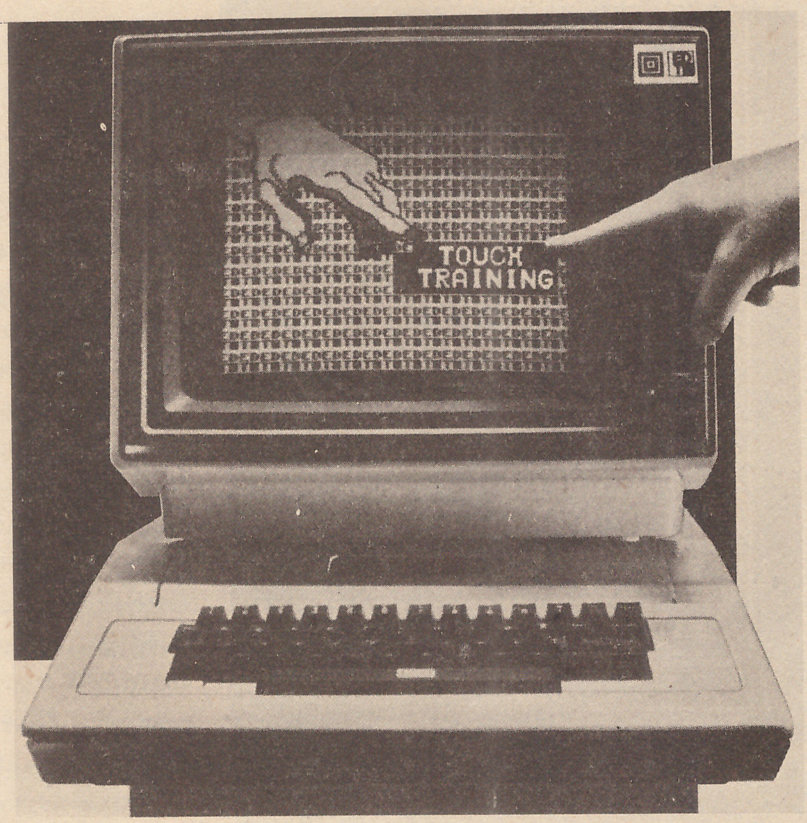
สำหรับในต่างประเทศแล้ว ศึกฮาร์ดแวร์ของไมโครคอมพิวเตอร์นับว่าใกล้จะยุติแล้ว คู่ท้าชิงคงเหลือเพียง 2 บริษัทยักษ์ใหญ่เท่านั้นที่ยังมีบทบาทในวงการซึ่งก็คือ IBM และ APPLE และดูเหมือนว่า IBM จะเป็นผู้ชนะในที่สุด ทั้งนี้เพราะ IBM PC ในปัจจุบันมียอดจำหน่ายแซง Apple II ซึ่งครองแชมป์มาตลอดระยะเวลาหลายปีที่ผ่านมา และเมื่อรวมกับเครื่องที่ Compatible กับ IBM PC ด้วยแล้ว ในปีนี้อาณาจักรของ IBM PC จะแผ่ขยายอิทธิพลครอบคลุมตลาดเกือบ 50% เข้าไปทุกทีแล้วและถ้าไม่มีอะไรมาหยุดยั้งการขยายตัวที่ว่านี้เป็นที่คาดกันว่าในเวลาเพียง 2 ปีอาณาจักรที่จะขยายไปถึง 75% โดย IBM PC เองจะมีส่วนแบ่งตลาดถึง 50% และก็ดูเหมือนว่าหน้าที่หยุดยั้ง IBM จะคงต้องตกอยู่กับ Apple อย่างหลีกเลี่ยงไม่ได้ ด้วยเหตุผลดังกล่าวนี้เอง Apple จึงได้ออก Macintosh ออกมาเพื่อเผชิญหน้ากับ IBM PC ซึ่งผลจะเป็นอย่างไรนั้นเป็นสิ่งที่น่าจับตามองเป็นอย่างยิ่ง

จากแนวโน้มต่าง ๆ ดังกล่าวข้างต้นในฉบับนี้เราจึงได้เสนอเรื่องราวทั้งของ IBM PC และ Macintosh เพื่อให้ท่านได้รับทราบข้อมูลอย่างทันต่อเหตุการณ์ IBM PC นั้นปัจจุบันได้รับการยอมรับแล้วว่าเป็นเครื่องไมโครฯ มาตรฐานสำหรับ 16 บิท เราจึงได้นำเรื่องราวโดยละเอียดของ IBM PC นี้มาเสนอแก่ท่านเป็นตอน ๆ อีกเรื่องหนึ่งที่คงจะเป็นที่สนใจของผู้ที่สนใจด้านฮาร์ดแวร์ก็คือเรื่องของไมโครโปรเซสเซอร์ 8086/8088 ซึ่งก็กำลังจะกลายเป็นมาตรฐานของ 16 บิท เช่นกัน และเช่นเดียวกันกับเรื่องของ MBASIC ซึ่งเขียนโดยไมโครซอฟท์ ซึ่งเป็นที่ยอมรับกับมานานแล้วว่าเป็นมาตรฐานเช่นกันผู้ที่สนใจหรือใช้งาน MBASIC อยู่คงจะสมใจที่จะได้คู่มืออ้างอิงในฉบับนี้

พบกันใหม่ฉบับหน้าครับ

เสริมพร พูลทรัพย์

พูดกับคอมพิวเตอร์



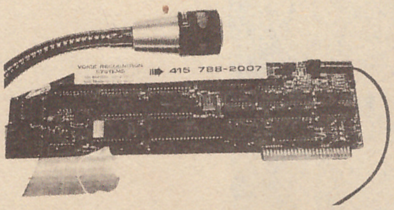
Apple Voice Input Module (AVIM) คือระบบจดจำเสียงพูดที่สามารถทำงานได้กับโปรแกรมทุกโปรแกรม มันจะทำงานคล้ายกับ Keyboard emulator ซึ่งทำให้ผู้ใช้สามารถป้อนคำสั่งและข้อมูลโดยใช้การพูดลงบนในไมโครโฟน ระบบนี้สามารถสร้างคำศัพท์ต่าง ๆ ขึ้นได้ภายในเวลาไม่กี่นาทีและบันทึกเก็บไว้ในแผ่นดิสค์สำหรับการใช้งานในอนาคต ผู้ผลิตคุยว่า AVIM สามารถจดจำศัพท์ต่างด้วยความแม่นยำถึง 98%

ระบบของ AVIM มีบอร์ดซึ่งมี RAM ขนาด 8 K ซึ่งสามารถเก็บคำสั่งได้ประมาร 80 คำสั่ง ไมโครโฟนและแผ่นดิสค์ซึ่งเก็บ Utility คำสั่งต่าง ๆ ของ Apple soft BASIC, Visicalc, Word Star etc นอกจากนั้นยังมี option สำหรับใช้ FM Wireless microphone สำหรับ remote control อีกด้วย ราคาของ AVIM ประมาณ 950 ดอลลาร์

คอมพิวเตอร์ระบบสัมผัส

"Touch-N – Know" คือชื่อของระบบคอมพิวเตอร์แบบใหม่ ที่ใช้ในการสอนที่ใช้ระบบสัมผัสใช้ได้กับเครื่อง Apple II plus และ IBM PC package นี้ราคา 2950 ดอลลาร์ สามารถใช้ควบคุม ข่าวสาร รูปภาพและข้อมูลบนแผ่นวิดิโอดิสค์

ฮาร์ดแวร์ของ Touch-N-Know ประกอบด้วย interface card และจอสีที่มีเนื้อที่บนจอ 32 แห่งที่ sensitive กับความร้อนจากนิ้วทำให้ใช้เป็นคำสั่งสำหรับ execute โปรแกรมได้ ระบบ program-authoring system จะช่วยให้ผู้ใช้เขียน menu, รูปภาพกราฟฟิคและข้อความต่าง ๆ ได้สนใจติดต่อได้ที่ Educational Dimensions Training, POB 126, Stamford CT 06904 USA.



จอภาพสำหรับ Apple II

คุณเคยสงสัยบางไหมว่าทำไม Apple จึงขายจอ Monitor III ให้กับผู้ใช้ Apple II ไม่ต้องสงสัยอีกต่อไปแล้วเพราะเดี๋ยวนี้ Apple ได้ออก Monitor II จอมอนิเตอร์ที่สามารถใช้ได้กับ Apple II , II plus และ II e

จอภาพนี้ราคา 229 ดอลลาร์

ข่าวคอมพิวเตอร์



เป็นจอสีแบบ high resolution ขนาด 12 นิ้ว แสดงตัวอักษรได้ 80 คอลัมน์ จอภาพปรับเอียงขึ้นลงได้ต่อกับ Apple II ได้และใช้สายเคเบิ้ลมาตรฐานของ Apple II

ยุคใหม่ของ Wordprocessor

Word เป็น wordprocessor ใหม่ล่าสุดจาก Microsoft สามารถแสดงข้อความบนจอได้เหมือนกับที่ต้องการให้พิมพ์ ซึ่งผิดกับ Word-processor ตัวอื่นเช่น Word Star ถ้าจะพิมพ์ตัวหนาต้องสั่ง ^B...^B หรือขีดเส้นใต้ต้องสั่ง^S.....^S ตรงข้อความที่ต้องการ ส่วน Word สามารถแสดงลักษณะตัวอักษรได้เหมือนกับที่จะถูกพิมพ์ลงบนเครื่องพิมพ์ได้เลย นอกจากนั้นคำสั่งต่าง ๆ ที่ใช้เช่นในการจัด paragraph จัด center ยังง่ายกว่ามาก นอกจากนั้นยังสามารถใช้ร่วมกับ Mouse ของ Microsoft ซึ่งทำให้การทำงานเร็วขึ้นมาก

คอมพิวเตอร์รุ่นยักษ์ใช้โปรเซสเซอร์แปดตัว

หลังจากที่เงียบหายไปนาน คอนโทรลดาต้าคอร์ปอเรชั่น (CDC) ผู้ประสบผลสำเร็จจากซูเปอร์คอมพิวไซเบอร์ 205 ก็ได้เปิดเผยแผนการเกี่ยวกับเครื่องคอมพิวเตอร์ขนาดยักษ์ที่ใช้โปรเซสเซอร์ถึง 8 ตัว คอมพิวเตอร์ที่ว่านี้ตั้งชื่อเรียกกันไว้ว่า 2xx จะประกอบขึ้นจากกลอจิกอาเรย์ชนิดคอมพลีเมนทารี-มอส (complementary-MOS logic arrays) แต่ละอาเรย์จะมีเกท (gate) 20,000 เกท หรือมากกว่านั้นภายใต้ระบบทำความเย็นด้วยไนโตรเจนเหลว ซึ่งเบ็ดเสร็จแล้วมีผลให้ความเร็วในการทำงานสูงกว่ารุ่น 205 ซึ่งเป็นคอมพิวเตอร์ที่เร็วที่สุดเท่าที่มีใช้ในตลาดถึง 40 เท่า CDC กล่าวว่าตัวเลขความเร็วที่สูงกว่า 40 เท่านี้ประมาณมาจากความเร็วของโปรเซสเซอร์แบบฟอนนิวมานน์(von Neumann)แต่ละตัวใน 8 ตัวที่ทำงานร่วมกันเป็น 2XX ซึ่งเร็วกว่าโปรเซสเซอร์เดี่ยวของ 205 ที่เร็วที่สุดในปัจจุบันถึง 5 เท่า อย่างไรก็ตามเราเองก็ยอมรับว่าด้วยเทคโนโลยีในปัจจุบันการพ่วงโปรเซสเซอร์หลายตัวให้ทำงานร่วมกันจะทำให้ขีดความสามารถตกลงบ้าง แต่เขาก็เชื่อว่ากว่าจะถึงปี 1986 ความก้าวหน้าทางซอฟท์แวร์จะยังผลให้สถาปัตยกรรมแบบโปรเซสเซอร์หลายตัวขนานกันทำงานได้มีประสิทธิภาพขึ้น

ทาง CDC ระวังถ้อยคำมากเมื่อจะระบุขีดความสามารถที่คาดไว้ของ 2XX เป็นตัวเลขแน่นอน เครื่องนี้จะทำลายสถิติความเร็วสูง "แน่นอนว่าเกิน 10 กิกะฟล็อป" (gigaflops = billion

ข่าวคอมพิวเตอร์

floating-point operation per second คือคำนวณตัวเลขทศนิยมได้พันล้านครั้งใน 1 วินาที)

เมื่อประกอบเต็มที่ 2XX มีโปรเซสเซอร์ย่อย 8 ตัว แต่ละตัวมีหน่วยความจำ 32 เมกะไบท์ (32 ล้านไบท์ ๆ ละ 8 บิท) และมีหน่วยความจำหลักที่ใช้ร่วมกันอีกถึง 2 กิกะไบท์ (2 พันล้านไบท์) ในขณะที่ 205 มีความเร็วสูงสุด 800 เมกะฟล็อป (คำนวณตัวเลขทศนิยมได้ 800 ล้านครั้งใน 1 วินาที) โดยทำความเร็วเฉลี่ยได้ 400 เมกะฟล็อป เมื่อมีหน่วยความจำหลักอยู่เต็มที่ 64 เมกะไบท์

หลังจากการปรากฏโฉมของไซเบอร์ 205 เมื่อกลางปี 1982 เครื่องคอมพิวเตอร์ที่ทำลายสถิตินี้ได้คือเครย์ X–MP ของเครย์รีเสอร์ซคอร์ปอเรชั่น ทำความเร็วได้ 420 เมกะฟล็อป อย่างไรก็ตามเมื่อเดือนเมษายนที่ผ่านมา เอ็นอีซีคอร์ปอเรชั่นแห่งโตเกียว ก็ประกาศจะวางตลาดเครื่องที่ทำความเร็วเฉลี่ยได้ 1.3 กิกะฟล็อปในราวต้นปี 1985 และยังมีบริษัทญี่ปุ่นอีก 2 เจ้า คือฮิตาชิและฟูจิตซึที่ประกาศเครื่องในระดับความเร็ว 500 ถึง 600 เมกะฟล็อป

กุญแจสำคัญในความสามารถ 2XX เห็นจะได้แก่การพัฒนา ซีมอส (C-MOS) อาเรย์ซึ่งเทียบเท่ากับเกท 20,000 ถึง 40,000 เกทได้สำเร็จ เพราะเทคโนโลยีปัจจุบันกับซีมอส สามารถบรรจุเกทได้สูงสุด 8,000 เกทบนชิพ (chip) 1 ชิพ ซึ่งโดยทั่วไปแล้วจะบรรจุเกทอยู่ในช่วง 2,000 ถึง 3,000 เกทเท่านั้น การที่ชิพมีความหนาแน่นของเกทสูงขึ้นจะช่วยลดเวลาที่เสียไปเมื่อสัญญาณต้องเดินทางระหว่างชิพ (เมื่อคอมพิวเตอร์มีความเร็วสูงถึงขนาด แม้เวลาที่อีเลคตรอนใช้เคลื่อนที่ในสายตัวนำก็มีผลขึ้นมาอย่างไม่มีทางเลี่ยง) เวลาที่เสียไปนี้เป็นขีดจำกัดสำคัญทางด้านความเร็วของซูเปอร์คอมพิวเตอร์ในปัจจุบัน

การที่ CDC เลือกใช้เทคโนโลยีแบบซีมอส สำหรับวงจรลอจิกของ 2XX นั้นมีคำอธิบายว่า "เพราะเราจะไปสู่การย่อส่วนขนาดใหญ่โดยเลี่ยงไม่ให้เกิดความร้อนสูงอย่างแน่นอน" (เป็นที่ทราบกันดีว่าสารกึ่งตัวนำประเภทโลหะออกไซด์หรือ metal oxide semiconductor จะกินเนื้อที่บนชิพน้อยมาก และมีอัตราสูญเสียกำลังต่ำ และยิ่งประกอบเป็นเกทโดยใช้วงจรแบบ complementary ด้วยแล้วอัตราสูญเสียกำลังจะยิ่งต่ำลงไปอีก ทำให้ความร้อนที่เกิดขึ้นเนื่องจากการทำงานลดต่ำลงเป็นเงาตามตัว แต่ข้อเสียที่สำคัญของซีมอสก็คือความเร็วซึ่งต่ำมาก) ยกตัวอย่างเช่น อาร์เรย์ซีมอส 1 ชิพเทียบเท่ากับเกท 20,000 เกทจะมีกำลังงานสูญเสียเพียง 1.5 ถึง 2 วัตต์ ในขณะที่อุปกรณ์ลอจิกประเภทอีมิตเตอร์คัปเปิล (emitter-coupled-logic เป็นอุปกรณ์ลอจิกที่เร็วที่สุด) 1 ชิพเทียบเท่ากับเกท 250 เกท ที่ใช้ในไซเบอร์ 205 มีกำลังสูญเสียถึง 4 วัตต์ เทคโนโลยีทำชิพโดยใช้สารแกลเลียมอาเซไนด์หรือสารอื่น ๆ ที่ทำงานเร็วมากนั้นก็มีกำลังสูญเสียสูงเกินกว่าจะนำมาย่อส่วนมาก ๆ ได้ การที่จะคั้นเอาความเร็วสูง ๆ จากอุปกรณ์ประเภทซีมอสซึ่งเป็นวงจรลอจิกที่ความเร็วต่ำกว่าใครเพื่อนนั้น CDC วางแผนไว้ว่าจะใช้ไนโตรเจนเหลวทำให้มันเย็นลงไปให้จุดทำงานของมันอยู่ที่อุณหภูมิประมาณ 77 องศาเคลวิน

การออกแบบ 2XX นั้นทำไว้ให้ซอฟท์แวร์ใช้แทนกับของ 205 ได้ แต่โปรแกรมของ 205 ก็คงจะไม่มีประสิทธิภาพดีเท่ากับโปรแกรมที่เขียนขึ้นเป็นพิเศษเพื่อใช้ข้อได้เปรียบของเครื่องใหม่ที่มีโปรเซสเซอร์ 8 ตัวให้เต็มที่

ห้ามใช้คอมพิวเตอร์บนเครื่องบิน

คุณอาจจะเคยเห็นโฆษณา Portable Computer ที่นักธุรกิจนำคอมพิวเตอร์ของตนการใช้งานของตนได้อย่างสนุกสนานแม้กระทั่งบนที่นั่งบนเครื่องบิน ถ้าคุณคิดว่าการเอาคอมพิวเตอร์ไปใช้บนเครื่องบินเป็นของโก้ละก็คุณคิดผิดถนัดเพราะสิ่งหนึ่งที่บรรดาโฆษณาไม่ได้บอกคุณก็คือบรรดาสายการบินส่วนใหญ่ได้ห้ามการใช้คอมพิวเตอร์และวิดีโอเกมส์บนเครื่องบินอย่างเด็ดขาด เพราะสัญญาณจากเครื่องดังกล่าวจะไปรบกวนกับเครื่องนำร่องและระบบสื่อสาร



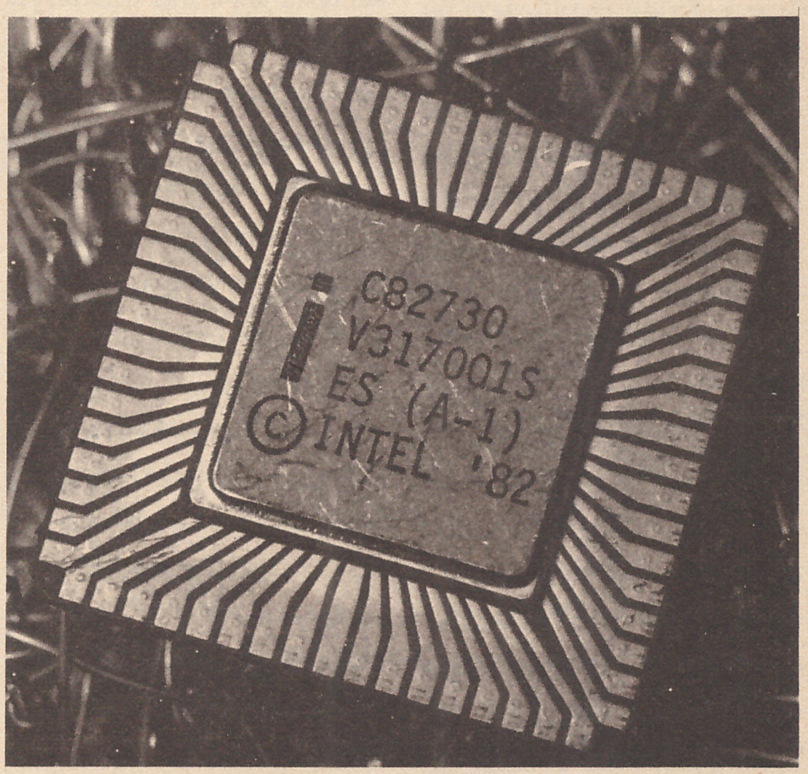
ซึ่งอาจจะทำให้การบินของเที่ยวนั้นไม่ปลอดภัยได้

ไมโครโปรเซสเซอร์ 32 บิท ศึกที่เพิ่งจะเริ่ม

ในขณะที่การขับเคี่ยวกันอย่างเอาเป็นเอาตาย กำลังดำเนินไปในตลาดของไมโครโปรเซสเตอร์ 16 บิท ก็ดูเหมือนเพิ่งจะเริ่มต้นเท่านั้น ยังไม่มีสิ่งใดส่งให้เห็นว่าใครจะเป็นผู้นำในตลาดระดับนี้อย่างแท้จริง

บริษัท ไซลอก **(Zilog)** ผู้ผลิต Z–80 (8 บิท) และ Z–8,000 (16 บิท) กำลังจะออก Z–80,000 ซึ่งเป็นไมโครโปรเซสเซอร์ขนาด 32 บิทที่ดูเหมือนจะก้าวหน้ามากตัวหนึ่ง ซึ่งสามารถจะอ้างถึงหน่วยความจำได้ (โดยใช้แอดเดรสบัสขนาด 32 บิท) เต็มที่ถึง 4 จิกะไบท์ (232 = 4,294,567,296 ไบท์) ทั้งโดยการแบ่งเป็น Segment และการอ้างแอดเดรสโดยตรง (ไมโครโปรเซสเซอร์ 8 บิทใช้หน่วยความจำได้แค่ 64 กิโลไบท์ หรือ 2=65,536 ไบท์เท่านั้น) นอกจากนี้ Z–80,000 ยังมีวงจรที่จำเป็นสำหรับการทำ virtual memory อยู่ภายในชิพ และระบบการ fetch และ execute คำสั่งหลาย ๆ คำสั่งโดยต่อเนื่องกันในจังหวะการทำงานเดียว หรือที่เรียกว่า pipelined ได้ถึง 6 ระดับ ทำให้สามารถทำงานได้ถึง 5 ล้านคำสั่ง/วินาที

โครงสร้างที่สนับสนุนการทำ pipelined ใน Z–80,000 ก็คือหน่วยความจำย่อยภายในชิพ (chache memory) ซึ่งจะเป็นที่ให้ CPU อ่านคำสั่งหรือข้อมูลที่อยู่ถัดไปจากที่กำลังใช้การ execute มาเก็บไว้ก่อน เมื่อจะใช้จริง ๆ ในการทำงานต่อไปจะได้ไม่เสียเวลาไปอ่านจากหน่วยความจำอีก ใน Z–80,000 จะมี chache memory สำหรับทั้งคำสั่ง (instruction) และข้อมูล (operand)



ข่าวคอมพิวเตอร์

ส่วนบริษัท **โมโตโรล่า** ผู้ผลิต MC 68000 ที่ใช้ในคอมพิวเตอร์รุ่นใหม่ของ Apple คือ Lisa และ Macintosh นั้นก็พัฒนาไมโครโปรเซสเซอร์เบอร์ใหม่คือ MC68020 ขนาด 32 บิท ซึ่งใช้ chache memory เหมือนกันแต่ใช้เฉพาะในการเก็บคำสั่งเท่านั้น เนื่องจากทางบริษัทมีโครงการจะพัฒนาไอซีพิเศษที่ทำงานร่วมกับไมโครโปรเซสเซอร์ หรือที่เรียกว่า Co-Processor จำนวนมากเช่น MC 6881 ซึ่งใช้คำนวณเลข floating point และมีความเห็นว่าการไม่ใช่chache memoryกับข้อมูลจะเหมาะสมกว่าในระบบที่มีการทำงานร่วมกับ Co-processor หลายตัวเพราะข้อมูลที่โปรเซสเซอร์ทุกตัวทำงานด้วยนั้นจะเชื่อถือได้มากกว่าเนื่องจากเป็นข้อมูลตัวเดียวกัน นอกจากนี้ยังใช้เทคนิคการผลิตแบบ n-channel และ Complementary MOS เพื่อพยายามจะลดกำลังงานสูญเสียใน 68020 ให้เหลือเพียง 1.5 ถึง 2 วัตต์ เท่า ๆ กับใน 68000 ซึ่งเป็นไมโครโปรเซสเซอร์ขนาด 16 บิท

ในขณะเดียวกัน **อินเทล** ซึ่งเป็นผู้นำในตลาด 16 บิทอยู่ขณะนี้ด้วยไมโครโปรเซสเซอร์ในตระกูล 8086/88, 80186 และ 80286 กำลังจะออก 80386 ซึ่งเป็นแบบ 32 บิทที่คาดว่าจะมีประสิทธิภาพสูงกว่า 80286 ถึงเท่าตัวและนอกจากจะมีระบบจัดหน่วยความจำแบบ Segment อย่างเดียวกับ 80286 แล้ว ยังรวมเอาระบบ virtual memory เข้าไปไว้ในชิพด้วยเช่นเดียวกับ Z–80,000 ของไซลอก และนอกจาก 80386 แล้วอินเทลยังมี iAPX–432 ซึ่งเป็นไมโครโปรเซสเซอร์แบบ 32 บิทอีกตัวหนึ่งที่ประกอบด้วย 3 ชิพ และมีชุดคำสั่งที่สนับสนุนภาษาคอมพิวเตอร์ Ada ด้วย (ดูรายละเอียด iAPX–432 ในไมโครฯฉบับที่ 3)

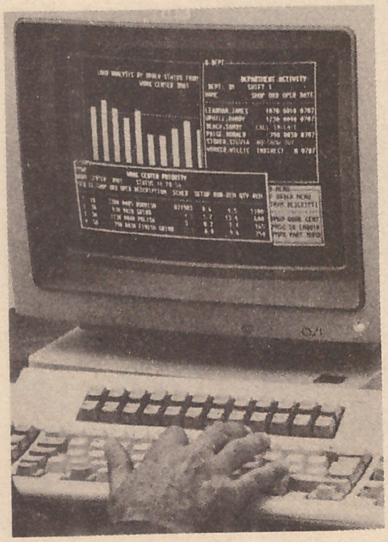
ผู้ผลิตไมโครโปรเซสเซอร์ที่สำคัญอีกรายหนึ่งคือ **เนชั่นแนล เซมิคอนดัคเตอร์** ก็กำลังจะออก 32132 ซึ่งเป็นแบบ 32 บิท ทำงานที่ความถี่ 20 MHZ ผลิตโดยวิธีการ 1.5 Micrometre CMOS process ประกอบด้วยทรานซิสเตอร์กว่า 500,000 ตัว โดยตามแผนที่วางไว้จะมีระบบ virtual memory แบบ demand paging, ระบบ floating point arithmatic ตามมาตรฐานของสถาบัน IEEE (Institute of Electrical and Electronic Engineering)

ถ้าจะพิจารณาถึงแผนการของแต่ละบริษัทแล้วจะเห็นว่าบริษัททั้ง 4 ต่างก็จะออกผลิตภัณฑ์ใหม่ของตนในช่วงสองปีข้างหน้านี้ทั้งสิ้น โมโตโรล่าจะออก 68020 ในต้นปี 1984 ไซลอกตามมาในฤดูหนาวปี 1984 อินเทลก็คาดว่าจะออกมาตอนปลายปีส่วน 32132 ของ NS นั้นคงจะยังไม่ออกมาเป็นชิพจนกว่าจะถึงปี 1985 แต่ NS ก็จะออก 32032 ซึ่งมีแอดเดรส 24 บิทเป็นตัวอย่างในปีนี้

ไม่ว่าบรรดาผู้ผลิตทั้งหลายจะเร่งรีบออกผลิตภัณฑ์ของตนอย่างไรก็ตาม ตลาดของไมโครโปรเซสเซอร์ 32 บิทก็ยังมีขนาดเล็กประมาณ 100,000 ตัวในปี 1984 และอาจขึ้นถึง 1 ล้านในปี 1985 ทั้งนี้ตามรายงานของบริษัทวิจัย Dataquest ดังนั้นจนกว่าจะถึงปี 1985 คงมีผู้นำไมโครโปรเซสเซอร์ 32 บิท ไปใช้ในงานต่าง ๆ เช่น multi-user, หุ่นยนต์, การทำภาพกราฟฟิคเป็นจำนวนไม่มากนัก

ไมโครเทอมินัล IBM 3270

IBM เริ่มขยายอาณาจักรไมโครของตนอีกครั้งโดยได้ออก 3270-PC ซึ่งเป็นไมโครที่ทำงานได้เป็นทั้ง IBM PC และเทอมินัล 3270 หรือทำงานทั้ง 2 แบบได้พร้อม ๆ กัน 3270-PC เป็นการรวมเอาจุดเด่นของ PC ที่มีโปรแกรมใช้งานและ utility ต่าง ๆ มากมายเข้ากับความสามารถในการประมวลผลของคอมพิวเตอร์เมนเฟรม



ข่าวคอมพิวเตอร์

3270-PC มีรูปร่างหน้าตาคล้ายกับ IBM PC จะแตกต่างกันก็ที่คีย์บอร์ดที่มี 122 คีย์ มี 24 ฟังชั่นคีย์ เหมือนกับเทอมินัล 3270 โค้ดของตัวอักษรใช้ EBCDIC ของ IBM แทน ASC II

จอภาพเป็นแบบ 5272 ขนาด 14 นิ้ว high resolution ขนาด 720 x 350 pixels มี 8 สี RAM มีขนาดตั้งแต่ 256 K ถึง 640 K floppy disk drive 1 หรือ 2 ตัว หรือ harddisk แบบ Winchester ขนาด 10 เมกกะไบท์

3270-PC สามารถติดต่อได้กับคอมพิวเตอร์ IBM ขนาดใหญ่ได้หลายแบบเช่น System/370 43 XX, และ 308 x series และ Operating System ทำให้ 3270-PC สามารถทำงานได้ถึง 7 งานในเวลาเดียวกัน โดยแสดงออกมาเป็น "window" บนจอภาพ และสามารถส่งผ่านข้อมูลระหว่าง window ได้

3270 จะวางจำหน่ายใน 3 แบบ คือ Model 2 มี 256 K RAM floppy disk drive 1 ตัว ราคา 4290 ดอลลาร์ Model 4 มี RAM ขนาด 320 K และ disk drive 2 ตัว ราคา 5319 ดอลลาร์ Model 6 มี 10 megabyte hard disk drive, floppy drive 320 K RAM ราคา 7180 ดอลลาร์ จอสีราคา 995 ดอลลาร์ Control program ราคา 300 ดอลลาร์และ File ที่ใช้ transfer program กับ host computer ราคา 600 ดอลลาร์

3270-PC จะออกวางจำหน่ายในตอนต้นปีนี้



เมื่อคอมพิวเตอร์เล่นหนัง

"Whiz Kids" คือชื่อของภาพยนตร์โทรทัศน์ของ CBS ชุดใหม่ที่เริ่มออกฉายในสหรัฐในเดือนกันยายนที่ผ่านมานี้ เป็นเรื่องของเด็กวัยรุ่นกลุ่มหนึ่งซึ่งหัวหน้ากลุ่ม คือ ริชชี ซึ่งมีความสามารถพิเศษในการใช้คอมพิวเตอร์ในการเฉลยปัญหาอาชญากรรมที่แม้แต่ตัวเราก็ยังสืบสวนไม่ได้

"ดารา" รับเชิญของภาพยนตร์เรื่องนี้แทนที่เป็นดาราดัง ๆ แต่กลุ่มเป็นคอมพิวเตอร์รุ่นดัง ๆ แทน เช่น Apple II , Lisa, Grid Compass หรือแม้กระทั่ง IBM Peanut ซึ่งจะเป็นกลไกสำคัญในการเฉลยปัญหาสำคัญ ๆ ของเรื่อง นอกจากคอมพิวเตอร์แล้วผู้อำนวยการสร้างของภาพยนตร์เรื่องนี้คือ Philip Deguere ได้เซ็นสัญญานำ HERO I หุ่นยนต์รับใช้ชื่อดังจากบริษัทมาเข้ากล้องกับเขาด้วย

เกมส์เลเซอร์ดิสค์

ยุคใหม่ของวิดีโอเกมส์ได้เริ่มต้นขึ้นแล้วด้วย "Dragon Lair" วิดีโอเกมส์ 3 มิติแบบใหม่ที่ใช้เลเซอร์ดิสค์ประกอบเข้ากับคอมพิวเตอร์ Dragon Lair สามารถสร้างเกมส์ที่เป็นการ์ตูน 3 มิติขึ้นบนจอภาพ ผู้เล่นสามารถ



บังคับกริยาท่าทางตลอดจนฉากการต่อสู้ของอัศวิน "Dirk the Daring" ผู้ซึ่งบุกไปยังปราสาทมังกรเพื่อช่วยเจ้าหญิง Daphe ผู้ออกแบบเกมส์นี้คือ Don Bluth ผู้ซึ่งเคยสร้างภาพยนตร์การ์ตูนให้กับ MGM/United Artists คือเรื่อง The Secret of NIMH

ชิพใหม่จากอินเทล

หลังจากประสบความสำเร็จอย่างดงามจาก ไมโครโปรเซสเซอร์ 8088 ซึ่งใช้เป็น CPU ในเครื่อง IBM PC ที่แพร่หลายที่สุดในระดับ 16 บิทในปัจจุบัน บริษัท อินเทล ผู้ผลิต 8088 ก็ยังได้ออกผลิตภัณฑ์ใหม่อีกหลายตัวซึ่งสามารถใช้งานร่วมหรือแทน 8086/8088 ที่ใช้อยู่เดิม ได้จะเพิ่มประสิทธิภาพในการทำงานของระบบให้สูงขึ้น เช่น

-80186/80188 เป็น version ใหม่ของ 8086/8088 ที่นำเอาอุปกรณ์ต่าง ๆ ที่สำคัญซึ่งเคยเป็นไอซีต่างหากจาก CPU บนแผงวงจรหลักเข้าไปในตัวชิพเลย เช่น counter/timer,DMA controller, interrupt controller ทำให้ลดเนื้อที่บนแผงวงจรหลักลงได้ และทำงานได้รวดเร็วขึ้นทั้ง ๆ ที่ใช้คำสั่งเดิมของ 8086/8088 ได้และยังมีชุดคำสั่งเพิ่มเติมสำหรับควบคุมอุปกรณ์ต่าง ๆ ที่เพิ่มเข้ามาอีกด้วย 80186/8188 บรรจุใน leadless package 64 pin (ดังภาพ)

-80286 กล่าวกันว่าเป็นไมโครโปรเซสเซอร์ที่มีประสิทธิภาพสูงมาก พัฒนาจาก 8086 และ 80186 โดยบรรจุทรานซิสเตอร์ถึง 150,000 ตัวลงในชิพ การผลิตใช้เทคโนโลยีของ HMOS-III 80286 สามารถอ้างแอดเดรสถึงหน่วยความจำได้ถึง 16 Megabyte ซึ่งมากพอ ๆ กับไมโครโปรเซสเซอร์ 68000 ของบริษัทโมโตโรล่า ที่ใช้กับเครื่องรุ่น Lisa ของ Apple 80286 มีความเร็วในการทำงานถึง 1.5 ล้านคำสั่งต่อวินาที(ซึ่งเร็วกว่า 8086 ถึงห้าหกเท่า) นอกจากนี้ยังมี 80386 ซึ่งกำลังอยู่ระหว่างการพัฒนา คาดว่าจะมีประสิทธิภาพสูงกว่า 80286 ขึ้นไปอีก

Co-processor หรือตัวประมวลผลช่วย ได้แก่ชิพที่ทำหน้าที่พิเศษเฉพาะอย่างในการจัดการกับข้อมูลแทน CPU ซึ่งในชุดของ 8086/8088 นี้มีหลายตัวเช่น **8087** ซึ่งเป็น Nemeric Data Processor (NDP) ที่ทำหน้าที่ช่วย CPU ในการคำนวณตัวเลขต่าง ๆ เช่น Floating Point ให้ได้ผลลัพธ์รวดเร็วขึ้นและไม่ต้องอาศัยโปรแกรมที่ยืดยาวอีกต่อไป และ 8087 นี้ก็ได้ถูกพัฒนาเป็น 80287 เพื่อให้ใช้กับ CPU ที่พัฒนาแล้วด้วย

นอกจากนี้ยังมี **82730** ซึ่งทำหน้าที่เป็น text processor สำหรับจัดการกับข้อมูลที่เป็นข้อความ (text) ได้อย่างมีประสิทธิภาพขึ้น เช่นสามารถใช้ตัวเอน (italic) ตัวยกกำลัง (superscript) หรือตัวพ่วง (subscript) และอื่น ๆ อีกมากและ 82720 ซึ่งเป็น graphic display controller ที่จะช่วยให้การแสดงภาพกราฟฟิคทำได้สะดวกขึ้น ลดขนาดของซอฟท์แวร์ที่จำเป็นลง และทำให้การทำงานรวดเร็วขึ้น เช่นการแสดงผลของโปรแกรมหลาย ๆ โปรแกรมพร้อมกับบนจอ (Multiple display หรือการจัด window) แบบที่ใช้ในเครื่อง Lisa ของ Apple เป็นต้น

อย่างไรก็ตาม ในตลาดปัจจุบันยังไม่มีซอฟท์แวร์ใด โดยเฉพาะอย่างยิ่งโปรแกรมควบคุมระบบที่จะสนับสนุนหรือใช้ข้อความสามารถของชิพเหล่านี้อย่างเต็มที่ แม้แต่แผงวงจรเพิ่มเติม (expansion board) ที่ใช้ 80826 กับ IBM PC ก็ทำได้เพียงแต่เพิ่มความ

ข่าวคอมพิวเตอร์

เร็วในการทำงานขึ้นเล็กน้อยเท่านั้น จำเป็นที่จะต้องใช้ซอฟท์แวร์ที่เขียนขึ้นใหม่ จึงจะใช้ข้อความสามารถของชิพเหล่านี้ได้เต็มที่ เช่น XENIX version ของ 80286 ซึ่งคาดว่าจะออกสู่ตลาดก่อนสิ้นปี 1983 (XENIX คือโปรแกรมควบคุมระบบ Multi User ที่บริษัทไมโครซอฟท์ ผู้เขียนโปรแกรม PC-DOS. อันเป็นโปรแกรมควบคุมระบบที่ IBM PC ใช้อยู่ในปัจจุบัน พัฒนาขึ้นจากโปรแกรมควบคุมระบบ UNIX ของห้องทดลองเบลล์) มีรายงานข่าวว่าบริษัทไมโครซอฟท์,ห้องทดลองเบลล์ และบริษัทอินเทลกำลังเจรจาเพื่อตกลงเรื่องนี้กันอยู่

ส่วนทางดิจิตอล รีเสิร์ช ผู้ให้กำเนิด CP/M ก็กำลังดัดแปลง CP/M ให้ใช้กับ 80286 ได้อยู่เช่นกันถึงแม้ข่าวจะแจ้งว่าความพยายามของดิจิตอลรีเสิร์ชไม่คืบหน้าไปมากนักก็ตาม

ไมโครเมนเฟรม IBM XT/370

IBM ได้ทลายขีดจำกัดระหว่างไมโครคอมพิวเตอร์และเมนเฟรมคอมพิวเตอร์ได้แล้ว โดยได้ออกไมโครคอมพิวเตอร์รุ่นใหม่ของตนคือ PC XT/370 ซึ่งจะออกวางจำหน่ายในสหรัฐประมาณกลางปีนี้

XT/370 ใช้บอร์ดฮาร์ดแวร์ในการจำลองการทำงานของคอมพิวเตอร์ system/370 ทำให้ไมโครเครื่องนี้สามารถใช้โปรแกรมของ S/370 ได้นอกเหนือไปจากโปรแกรมที่ทำงานภายใต้ MS-DOS และ CP/M-86

เมื่อมองจากภายนอก XT/370 จะไม่แตกต่างจาก PC XT แต่ภายในจะมีบอร์ดเพิ่มขึ้นมา 3 บอร์ด บอร์ดแรกจะมีไมโครโปรเซสเซอร์ 3 ตัว ด้วยกัน คือ Motorole 68000 2 ตัว ทำงานด้าน binary operation code และ interger arithmatic และ Intel 8087 อีก 1 ตัว สำหรับงานคำนวณด้าน floating-point

บอร์ดที่ 2 จะบรรจุ RAM ขนาด 512 K เพื่อเพิ่มหน่วยความจำให้มากขึ้นนอกเหนือไปจาก 256 K ที่อยู่ในบอร์ดแม่อยู่แล้ว บอร์ดที่ 3 จะเป็นวงจรที่ทำให้ XT/370ทำหน้าที่เป็น 3277 terminal เมื่อต่อกับคอมพิวเตอร์เมนเฟรม

นอกจากฮาร์ดแวร์ที่แตกต่างกันแล้ว IBM ยังได้ดัดแปลง Operating System ของ 370 คือ VM/CMS (Virtual Machine/Conversational Monitor System) โดยเรียกชื่อว่า VM/PC โปรแกรมที่ทำให้ XT/370 ทำงานได้เป็น Workstation ของ System/370 โดยมี Virtual memory ถึง 4 megabyte ผู้ใช้ XT/370 สามารถเรียกโปรแกรมได้เกือบทุกโปรแกรมที่ทำงานภายใต้ VM/CMS มาทำงาน



ได้โดยไม่เกี่ยวข้องกับคอมพิวเตอร์เครื่องใหญ่ นอกจากนั้นผู้ใช้ยังสามารถเขียนโปรแกรมใหม่, ไฟล์และทำงานและส่งกลับไปยัง S/370 เพื่อ run หรือเก็บไว้ก็ได้ ในขณะเดียวกันก็จะสามารถสวิชท์กลับไปใช้ในโหมดของ IBM PC XT เพื่อทำงานโปรแกรมแบบ single user ได้ทุกเวลา

ระบบพื้นฐานของ XT/370 ประกอบด้วยการ์ด S/370 3 การ์ด, RAM ขนาด 256 K, floppy disk drive ขนาด 360 K 1 ตัว และ fixed disk ขนาด 10 หรือ 20 เมกกะไบท์ราคาขายในสหรัฐอยู่ในช่วง 8995 ถึง 11,690 ดอลลาร์ ผู้ที่ใช้ IBM PC XT อยู่สามารถซื้อชุด S/370 Kit เพื่อ upgrade เครื่องได้ในราคา 3790 ดอลลาร์ VM/PC Operating System ขายในราคา 1,000 ดอลลาร์

วิดีโอเทปสอนไมโครฯ

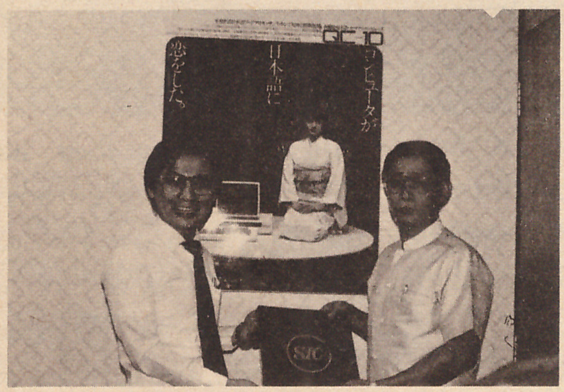
Comp Man Group เป็นชื่อของทีมงานคนหนุ่มไฟแรง ทีมแรกที่ได้นำเอาวิดีโอเทปเผยแพร่วิทยาการทางด้านไมโครคอมพิวเตอร์ เทปที่ทีมงานนี้จัดทำขึ้นมา มี 3 ม้วนด้วยกัน เป็นการใช้งาน (appl cation) โปรแกรม3 ตัว คือ dBASE II, Eord Star และ Supercalc เทปชุดนี้ลงทุนถ่ายทำอย่างประณีต แม้กระทั่งคนพากษ์ก็ยังใช้ทีมงานจากนักพากษ์ช่อง 3 จึงเป็นผลงานอีกชิ้นหนึ่งที่ผู้สนใจไมโครฯ และมีวิดีโอเทปอยู่แล้วน่าจะให้การสนับสนุน

ข่าวคอมพิวเตอร์

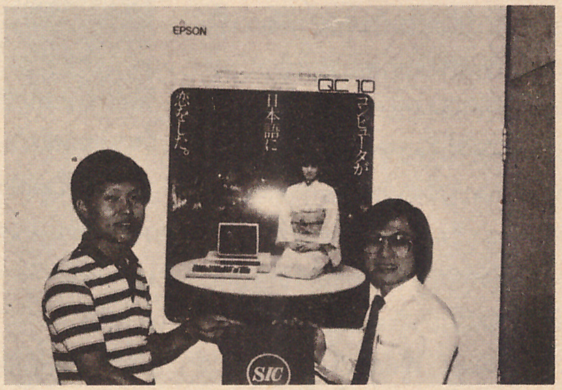
สหวิริยาตั้งตัวแทนใหม่

บริษัทสหวิริยาอินเตอร์เนชั่นแนลคอมพิวเตอร์ ได้แต่งตั้งตัวแทนจำหน่าย EPSON QX-10 เพิ่มเติมขึ้นอีก 4 แห่ง คือ

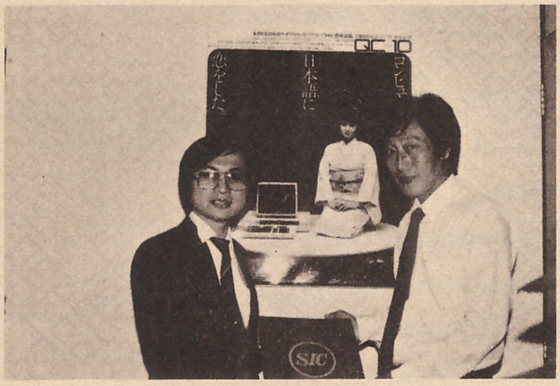
1. จังหวัด ชลบุรี บริษัท ไทย อนันท์คอมพิวเตอร์จำกัด มีคุณสัมฤทธิ์ ผลิตวานนท์ เป็นผู้จัดการ
2. จังหวัด จันทบุรี ห้างหุ้นส่วน จำกัด มุยฮวด คอมพิวเตอร์ มีคุณวีระ สงวนเศรษฐกิจ เป็นผู้จัดการ
3. จังหวัด สมุทรปราการ ได้แก่บริษัทปากน้ำอินเตอร์แนชนั่นแนลคอมพิวเตอร์ จำกัด มีคุณวิบูลย์ วัฒนาดิลกกุล เป็นผู้จัดการ
4. กรุงเทพมหานคร ได้แก่บริษัท นิวโรเท็ค จำกัด มีอ.นัยพินิชคชภักดี เป็นผู้จัดการ



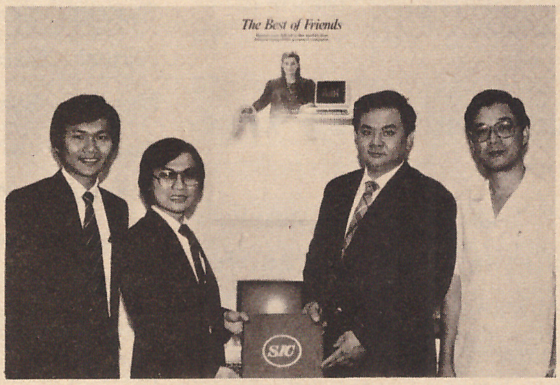
ชลบุรี ไทยอนันท์ คอมพิวเตอร์



จันทบุรี หจก. มุยฮวด คอมพิวเตอร์



สมุทรปราการ ปากน้ำอินเตอร์แนชั่นแนล



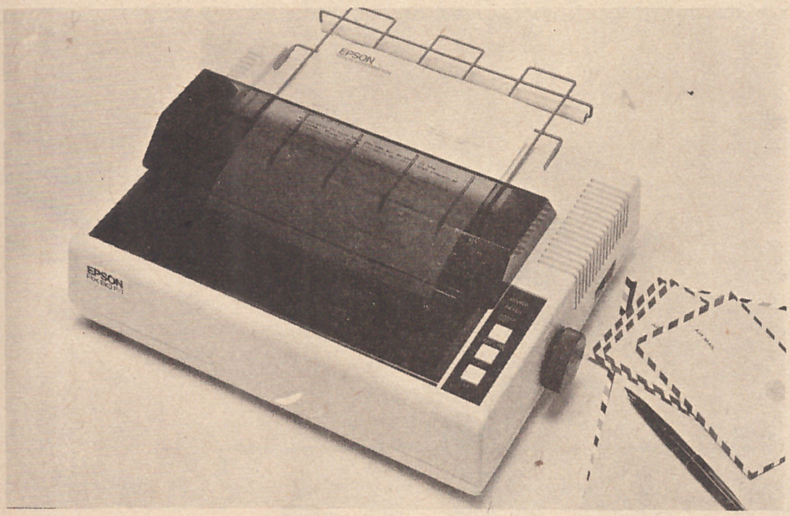
กรุงเทพฯ บริษัท นิวโรเท็ค จำกัด

สหวิริยา ช่วยกิจการตำรวจ

เมื่อวันที่ 13 มกราคม 2527 บ.สหวิริยาอินเตอร์แนชชั่นแนล คอม-พิวเตอร์ จำกัด ได้มอบเครื่องคอมพิวเตอร์ Epson QX-10 และ HX-20 ให้แก่ทางกรมตำรวจ เพื่อช่วยงานในหน่วยงานราชการกรมตำรวจ โดยทางบ.สหวิริยาฯ มีคุณ JACK (มิน ชุน ฮู) ผู้จัดการทั่วไปเป็นผู้มอบเครื่องให้แก่อธิบดีกรมตำรวจ พลตำรวจเอกณรงค์ มหานนท์



ข่าวคอมพิวเตอร์

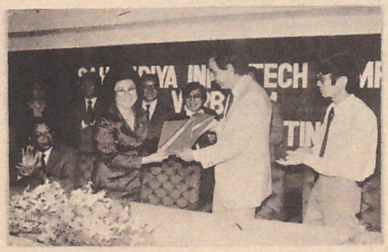


เครื่องพิมพ์ใหม่จาก EPSON

RX-80 FT เป็นเครื่องพิมพ์ราคาประหยัดรุ่นใหม่จาก EPSON ที่มีทั้ง Tractor feed และ Friction feed ทำให้สามารถใช้กระดาษได้ทั้ง 2 แบบ คือกระดาษแบบมีรูหนามเตยและกระดาษธรรมดาทำให้สะดวกในการใช้งานยิ่งขึ้น RX-80 F/T สามารถพิมพ์ได้เร็ว 100 ตัวอักษรต่อวินาที และยังมี mode ที่พิมพ์ที่ความเร็ว 50 ตัวอักษรต่อวินาทีเพื่อให้เสียงเงียบลงอีกด้วย

สหวิริยาได้เป็นตัวแทน Datalife

เมื่อวันที่ 27 กุมภาพันธ์ ศกนี้ ได้มีพิธีเซ็นสัญญา เป็นตัวแทนจำหน่ายผลิตภัณฑ์ Datalife ระหว่าง บริษัทสหวิริยาอินเตอร์แนชั่นแนล คอมพิวเตอร์กับบริษัท Verbatim ซึ่งเป็นผู้ผลิต floppy disk ที่ใหญ่ที่สุดในโลก



ในภาพคือ ดร.ประภา วิริยประไพกิจ ประธานบริษัทสหวิริยาอินเตอร์แนชชั่นแนล คอมพิวเตอร์กับ Mr. Richard Kaan ผู้จัดการทั่วไปทางภาคพื้นออสเตรเลียและภาคพื้นเอเชียของ Verbatim และ Mr. George Tham ฝ่ายจัดการฝ่ายการตลาดของ Verbatim

AP-200 Disk drive รุ่นบางเฉียบสำหรับ Apple

ผู้น่าสน Apple ไม่ว่าจะจริงหรือปลอมคงจะเคยประสบปัญหากับการใช้ Disk drive ราคาถูกมาบ้างไม่มากก็น้อย บัดนี้ Epson ได้เสนอทางเลือกใหม่ให้กับผู้ที่เล่น Apple แล้วด้วยการออก Disk drive สำหรับ Apple คือรุ่น AP-200 Ultraslim Floppy Disk Unit ซึ่งเป็นชนิด 2 Drive ขนาด $5\frac{1}{4}''$ มีความยาวพอที่จะวางบนเครื่อง Apple II หรือ II e แล้วตั้งจอภาพซ้อนได้อย่างไม่ยากเย็นนัก

ข้อเด่นของ AP-200 อยู่ตรงนี้ การทำงานที่เงียบสนิท ใช้งานได้สะดวกเพราะใช้ระบบป้อนดิสค์แบบกอปุ่มซึ่งสะดวกในการใช้งานมาก ตลอดจนชื่อเสียงของ Epson ในด้านการควบคุมคุณภาพก็เป็นประกันอยู่แลว สำหรับราคานั้น AP-200 (2 Drive) ราคาประมาณ 16,900 บาทซึ่งแพงกว่า Super-s เล็กน้อย แต่ก็นับว่าถูกกว่า Disk drive ของแท้ของ Apple มาก

แนะนำ SIC THAI DATABASE MANAGEMENT SYSTEM

บริษัทสหวิริยาอินเตอร์แนชชั่นแนล คอมพิวเตอร์ จำกัด เป็นบริษัทขายคอมพิวเตอร์หนึ่งในไม่กี่บริษัทที่ทุ่มเทให้กับการพัฒนาซอฟท์แวร์ภาษาไทยอย่างแท้จริงผลงานโปรแกรมมีสำเร็จภาษาไทยกว่า 20 โปรแกรม คงจะเป็นข้อพิสูจน์ในเรื่องนี้ได้เป็นอย่างดี

โปรแกรม THAI DATABASE เป็นโปรแกรมหนึ่งซึ่งทางบริษัทสหวิริยาฯ ได้พัฒนาขึ้นมาเองโดยไม่ได้มีการดัดแปลงหรือแก้ไขจากโปรแกรมสำเร็จรูปจาก

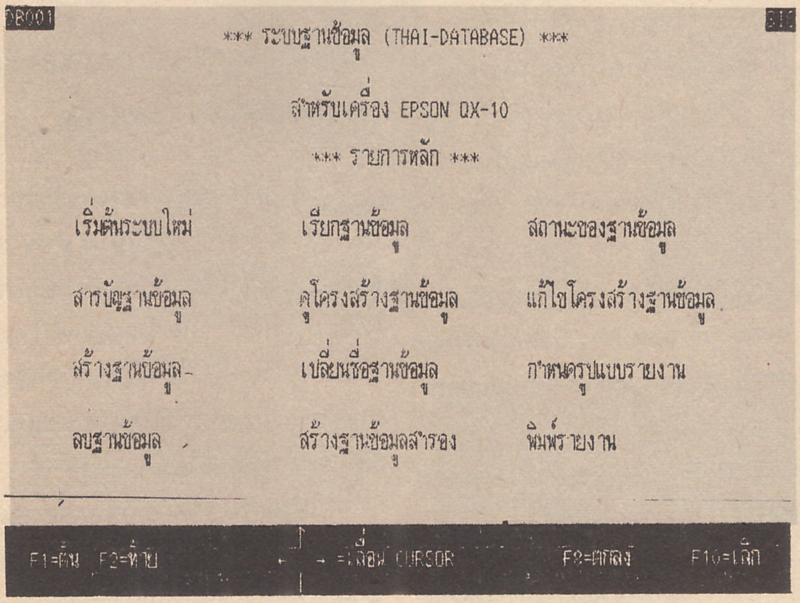
**** ที่อยู่ DEALER ****

รหัส	ชื่อ	บริษัท	ที่อยู่	โทรศัพท์
01	สมสิน ลีลอริคดิ์	รัตนผล	102-112 ถ.ท่าแพ อ.เมือง จ.เชียงใหม่	053-236767,234812
02	ภูเบศ แซ่ลิ้ม	VIDEO LAB	46 ถนนผัดพรัตนากร (ตลาดใหม่)อ.เมือง จ.เชียงใหม่	074-245075
03	กาญจนี	ACTION COMPUTER	5/26 ถนนศาลาแดง สีลม กทม	2335274-5
04	ชรวพล ศิริรัฐ	COMPUTER GROUP	786 ถ.โพธิ์กลาง อ.เมือง จ.นครราชสีมา	044-242170,244583
05	ไพบูลย์ ศักรพันธ์	ไทยพิมพ์สัมพัส	132/21 พระราม 6 กทม	2783457-8,2792317
06	ศิริชัย มุ่งจะกุล	ฟิลอสตรีเอพีนโพรดักส์	1895/101 ถ.พหลโยธิน บางเขน กทม.	5795822
07	ทวีช กิตติกรี	สหคอมพิวเตอร์	163-165 ถ.บรมไตร อ.เมือง จ.พิษณุโลก	055-259918

ตัวอย่างข้อมูลใน DATABASE

ต่างประเทศเลย นับว่าเป็นความริเริ่มที่น่าภูมิใจทีเดียวสำหรับนักพัฒนา software ภาษาไทยทีมนี้

โปรแกรม THAI DATABASE นี้มีจุดเด่นอยู่หลายประการ อาทิเช่น



Menu ของ SIC THAI DATABASE

- *ใช้งานได้ง่ายเพราะเป็น* **Menu Driven Program**
- *มี* **Function** *หลัก ๆ มากพอต่อการใช้งาน*
- *อาศัยประโยชน์ของ* **Function Key** *ต่าง ๆ ในการโต้ตอบกับโปรแกรม ทำให้สะดวกในการใช้งาน*
- *ตัวหนังสือใน 1* **record** *มีได้ถึง 1000 ตัวอักษร นับว่าใหญ่พอที่จะจัดการข้อมูลใหญ่ได้*
- **field** *ใน 1* **record** *มีได้ถึง 50* **field** *และมีส่วนเป็น* **Index field** *ได้ถึง 8* **field**
- *สะดวกในการกำหนดรูปแบบรายงาน ได้ทั้งทางจอภาพและเครื่องพิมพ์*
- *การเพิ่ม, ลบ, แก้ไข ข้อมูลตลอดจนการค้นหาข้อมูลในฐานข้อมูล ทำได้อย่างมีประสิทธิภาพ*
- *จำนวน* **record** *ต่อ 1 ฐานข้อมูลมีได้ถึง* **65535 records**
- *ตัวเลขใช้ได้ถึง 16 หลัก*
- *แผ่นข้อมูล 1 แผ่น จุฐานข้อมูลได้ 15 ฐานข้อมูล*
- *รายงานต่อฐานข้อมูลได้สูงสุด 15 รายงาน*

26 คำถามกับ Macintosh!

Macintosh เป็นไมโครคอมพิวเตอร์รุ่นล่าสุดของบริษัท Apple Computer ที่ได้รับความสนใจอย่างกว้างขวาง เนื่องจากมีราคาถูก และความสามารถสูงเช่นเดียวกับ Lisa ต่อไปนี้ เป็นคำถามคำตอบที่บริษัท Apple ได้จัดทำขึ้นแจกในการสัมมนาที่สิงคโปร์ เพื่อให้ เราสามารถทำความเข้าใจกับเจ้า "Mac" ขึ้นดียิ่งขึ้น

1 ทำไม **Apple** *ถึงเลือกไมโครโพรเซสเซอร์ชิพ 68000 ของ* **Motorola** *แทนที่จะเป็น 8088 หรือ 8086 ซึ่งเป็นที่รู้จักกันดีอยู่แล้ว*

■ Macintosh เป็นหนึ่งในสมาชิกของไมโครฯ ตระกูล 32 บิทของ Apple ไมโครโพรเซสเซอร์ 16 บิท ดังกล่าวมีความสามารถทางด้านความเร็ว กราฟิคและการ integrate โปรแกรม น้อยเกินไปที่สามารถรองรับเทคโนโลยีของ Lisa ได้ นอกจากนั้น Motorola 68000 ยังสามารถทำให้ Macintosh ใช้ software ของ Lisa ได้

68000 ใน Macintosh ทำงานที่ความถี่ 8 MHz ควบคู่ไปกับการทำงานของ ROM ขนาด 64 K

2 ทำไมถึงเลือก disk drive ขนาด $3\frac{1}{2}$ นิ้ว

■ disk drive ขนาดเล็กมีประโยชน์หลายประการ

— ประการแรก การออกแบบของ disk เหล่านี้อยู่ในระดับ state-of the-art และเป็นแนวโน้มของเทคโนโลยีในอนาคต

— ประการที่ 2 แผ่นดิสค์หน้าเดียวสามารถจุข้อมูลได้ถึง 400 K และมีความเชื่อถือ (reliability) ได้สูงมาก

— ประการสุดท้าย ดิสค์ไม่เพียงแต่ถูกบรรจุอยู่ในกล่องพลาสติคแข็ง ซึ่งมีความทนทานมากเท่านั้นแต่มันยังมีขนาดเล็กพอที่จะเอาใส่ในกระเป๋าเสื้อได้อีกด้วย! ปัจจุบันดิสค์ขนาด $3\frac{1}{2}$ นิ้วมีผู้ผลิตออกมาจำหน่ายแล้วหลายบริษัท (อ่านไอโครฯฉบับที่แล้วในคอลัมน์ข่าวคอมพิวเตอร์)

3 Macintosh ต่อ hard disk ได้ไหม

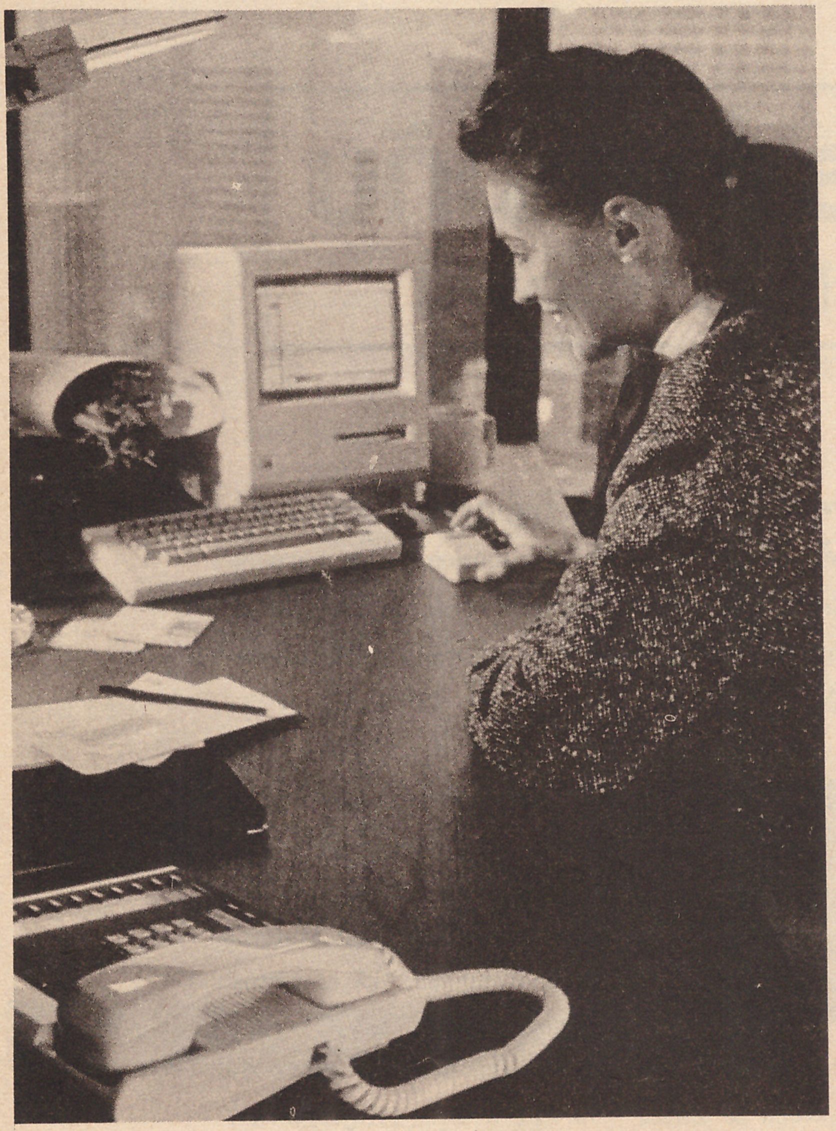
■ "Mac" ต่อ hard disk ได้โดยต่อสายเข้าทาง Apple Bus/Serial port ปัจจุบันบริษัท Apple ยังไม่ได้ออก hard disk สำหรับ Mac อย่างไรก็ตาม เนื่องจาก Apple เปิดเผย hardware ของ Mac แก่บริษัททั่วไป ก็เป็นที่คาดว่ามีบริษัทอื่น ๆ (third party) ที่กำลังออกแบบ hard disk ใช้กับ Mac อยู่แล้วจะนำออกจำหน่ายก่อนกลางปีนี้

4 จอภาพ Bit-mapped คืออะไร

■ Bit-mapping หมายถึงการที่จุดทุก ๆ จุดบนจอภาพมีตำแหน่งบิทอยู่ในหน่วยความจำด้วย การจัดความสัมพันธ์ของจอภาพและหน่วยความจำในลักษณะนี้ ทำให้ได้ภาพกราฟิคที่มีคุณภาพสูงบนจอ ทั้งนี้เพราะเราสามารถควบคุมตำแหน่งและขนาดของจุดบนจอภาพได้โดยตรงจอของ Mac สามารถแสดงจุดบนจอภาพได้ถึง 175,000 จุด ซึ่งจอภาพธรรมดาทำไม่ได้อีกประการหนึ่ง จอภาพแบบนี้ทำให้สะดวกในการเคลื่อนย้ายรูปภาพบนจอ เพราะเราเพียงแต่ "mapped" รูปจากตำแหน่งหนึ่งไปยังอีกตำแหน่งหนึ่งเท่านั้น

5. ทำไมถึงเลือกจอ 9 นิ้ว

■ สาเหตุที่เลือกเพราะต้องการขนาดเครื่องที่กะทัดรัด กินที่น้อยและเคลื่อนย้ายได้ง่ายเนื่องจากจอภาพของ Mac เป็นจอ bit-mapped high resolution อยู่แล้ว จึงมีความคมชัดสูงมาก ทำให้อ่านภาพและตัวหนังสือบนจอได้ง่ายพอ ๆ กับจอขนาดใหญ่ นอกจากนั้น Mac ยังใช้เทคนิคของ Window และการ Scroll ทำให้ไม่ต้องใช้จอใหญ่มากนักในการแสดงภาพ ทำให้ประหยัดเงินค่าจอไปได้มาก



รูปที่ 1 Macintosh มีขนาดเล็กกินเนื้อที่เพียงมุมของโต๊ะทำงานเท่านั้น



รูปที่ 2 สตีเวนจ็อบกับ Macintosh ในงานแถลงข่าวแก่ dealer ที่ชิคาโก

อย่างไรก็ตามก็กำลังมีผู้พัฒนาจอภาพที่ใหญ่กว่านี้ให้สำหรับ Macintosh อยู่ซึ่งคงจะได้เห็นโฉมกันในเร็ว ๆ นี้

*6. **Mac** จะมีจอสีไหม*

■ จุดประสงค์ของการออกแบบ Mac ก็เพื่อที่จะให้มีจอภาพขนาดเล็กที่มีความคมชัดสูงดังได้กล่าวมาแล้ว การที่จะทำให้มีจอภาพสีคุณภาพสูงได้นั้น จะทำให้ราคาของ Mac สูงกว่าราคาปัจจุบันเกือบ 2 เท่า

แต่อย่างไรก็ตามระบบ graphic ของ Mac สามารถที่จะใช้กับการออกจอสีจึงสามารถพัฒนาอุปกรณ์ output เป็นสีเช่น จอสี, plotter สี และ printer สี ได้ในอนาคต

*7. คีย์ควบคุม **Cursor** แบบ 4 ทิศทางใช้ได้กับ **Mac** หรือไม่*

■ สามารถใช้ได้แต่โปรแกรมของ Mac ส่วนใหญ่ไม่ต้องการมันเนื่องจากสามารถใช้ Mouse เลื่อน Cursor ทำให้การใช้งานสะดวกและรวดเร็วกว่ามาก

*8. ทำไมไม่มี **Numeric Keypad** บนคีย์บอร์ด*

■ การใช้งานส่วนใหญ่ไม่จำเป็นต้องใช้ การไม่ได้ใส่ไว้ทำให้คีย์บอร์ดมีขนาดเล็กลงและราคาถูกลง สำหรับงานที่จำเป็นต้องใช้จริง ๆ แล้ว เรามักจะต้องตั้ง keypad ไว้ในตำแหน่งที่เราใช้สะดวกที่สุดและก็มักจะไม่ใช้ตำแหน่งเดียวกับคีย์บอร์ด keypad จึงน่าจะเป็น option สำหรับการใช้งานมากกว่า

9. จะมีตัวอักษรแบบใหม่ ๆ บ้างไหม นอกเหนือจากตัวอักษรมาตรฐานบนจอ

■ มี ทั้งนี้เนื่องจาก ROM ของ Mac มีขนาดใหญ่จึงสามารถจุตัวอักษรได้หลายแบบรวมทั้งตัวอักษรที่ไม่ใช่ภาษาอังกฤษ และเครื่องหมายสัญลักษณ์ทางวิทยาศาสตร์ต่าง ๆ และธุรกิจอื่น ๆ อีกด้วย

*10. มีประโยชน์อะไรจากการใช้ **mouse** ปุ่มเดียว*

■ มีการวิจัยกันอย่างขนานใหญ่ในการออกแบบตัว mouse ทั้งนี้เพื่อเลียนแบบการทำงาน ของคน เช่น การชี้และเลือก โดยอาศัย mouse ปุ่มเดียว ทำให้ไม่ต้องเสียเวลาไปจำว่าปุ่มไหนใช้สำหรับอะไร เหมือนกับการใช้ mouse แบบอื่น ๆ ที่มี 2 หรือ 3 ปุ่ม

ยิ่งกว่านั้น mouse ของ Macintosh (และ mouse ของ Apple ทุกรุ่น) สามารถใช้งานได้กับพื้นแข็ง โดยใช้ลูกบอลวางเป็นตัวเลื่อน ไม่ต้องใช้พื้นรองแบบพิเศษเหมือน mouse แบบอื่น ๆ

ในสภาวะปกติ mouse จะสามารถ "วิ่ง" ไปได้หลายร้อยไมล์ ก่อนที่จะเอามาตรวจซ่อมสักครั้งหนึ่ง ปกติแล้วการบำรุงรักษาต้องเพียงแต่การทำความสะอาดลูกยางเท่านั้น

*11. ทำไม **Macintosh** จึงมี **ROM 64 K***

■ ROM 64 K นี้บรรจุ routine ใช้งานสำคัญ ๆ ของ Lisa ไว้ส่วนใหญ่ ทำให้ไม่ต้องเสียเวลาไปค้นหาจาก disk นอกจากนั้นเนื่องจาก code ต่าง ๆ ใน ROM สามารถทำงานได้เร็วกว่าใน RAM ถึง 25% ทำให้การ load โปรแกรมจาก disk เร็วขึ้นกว่าเดิม และยิ่งไม่ต้องเปลืองเนื้อที่สำหรับ routine ต่าง ๆ เหล่านี้ใน RAM ทำให้มีเนื้อที่เก็บข้อมูลใน

RAM มากขึ้นอีกด้วย

Mac มี routine ต่าง ๆ เหล่านี้มากมายอันเป็นผลมาจากเทคโนโลยีของ Lisa ทำให้ผู้ที่พัฒนา Software สามารถใช้เทคนิคชั้นสูงต่าง ๆ เช่น Window, pull down menu, cut และ paste integration ได้อย่างง่ายดาย โดยไม่ต้องเขียน routine เหล่านี้ใหม่ ทำให้ลดเวลาในการพัฒนา Software ลงไปได้มาก

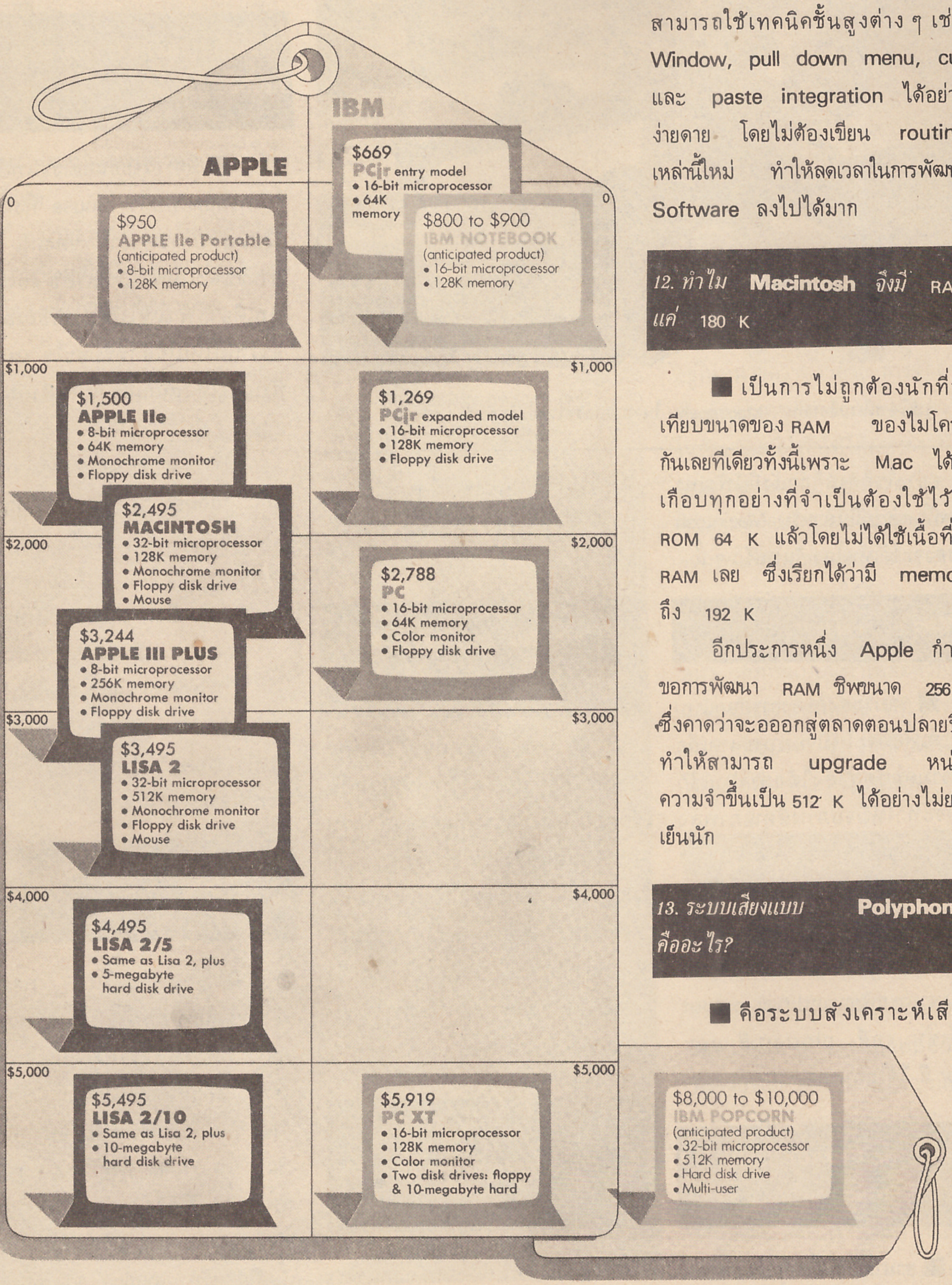
12. ทำไม **Macintosh** *จึงมี* RAM *แค่* 180 K

■ เป็นการไม่ถูกต้องนักที่จะเทียบขนาดของ RAM ของไมโครฯ กันเลยทีเดียวทั้งนี้เพราะ Mac ได้ใส่เกือบทุกอย่างที่จำเป็นต้องใช้ไว้ใน ROM 64 K แล้วโดยไม่ได้ใช้เนื้อที่ใน RAM เลย ซึ่งเรียกได้ว่ามี memory ถึง 192 K

อีกประการหนึ่ง Apple กำลังขอการพัฒนา RAM ชิพขนาด 256 K ซึ่งคาดว่าจะอออกสู่ตลาดตอนปลายปีนี้ ทำให้สามารถ upgrade หน่วยความจำขึ้นเป็น 512 K ได้อย่างไม่ยากเย็นนัก

13. ระบบเสียงแบบ **Polyphonic** *คืออะไร?*

■ คือระบบสังเคราะห์เสียง



รูปที่ 3 แนวรบของ Apple ที่จะหยุดยั้งการโจมตีทางการตลาดของ IBM

หลายเสียง ซึ่งเป็นระบบที่นับว่าก้าวหน้าที่สุดในปัจจุบันนี้ ระบบนี้สร้างไว้ในตัว Macintosh เรียบร้อยแล้ว ระบบนี้ใช้ digital analog converter ขนาด 8 bit ด้วย Sampling rate ขนาด 22 KHz สังเคราะห์เสียงได้ 4 เสียงพร้อม ๆ กัน โดยมี resolution ถึง 24 bit pitch ตลอดจน wave form ที่ผู้ใช้จะสามารถ define ได้

14. **Programer** *สวิชท์ คืออะไร?*

■ คือสวิชท์ที่ผู้ใช้สามารถที่จะ reset หรือ cold boot ระบบได้โดยไม่ต้องปิด power supply ทำให้สะดวกโดยเฉพาะสำหรับโปรแกรมเมอร์ที่ต้องใช้งานนี้บ่อย ๆ นอกจากนี้ยังมีปุ่ม NMI (Non Mistake Interrupt) สำหรับ start โปรแกรม debugger อีกด้วย

15. ทำไมจึงไม่มี **Expansion Slot** *ใน* **Mac**

■ Mac มีระบบทุกอย่าง build-in อยู่แล้วเช่น display, disk drive, serial port และ speech and sound hardware

Peripheral ต่าง ๆ ก็สามารถต่อเข้ากับ Mac ได้โดยตรง เช่น external drive, mouse, audio output, printer, modem ก็มี port ของแต่ละอย่างโดยเฉพาะจึงไม่จำเป็นที่จะเพิ่มเติมอะไรอีก

Apple เชื่อว่าการพัฒนาเครื่องของ Mac ให้ใช้งานได้ดีขึ้นกว่านี้ในอนาคตนั้นมาจากการพัฒนาทาง software โดยอาศัย ROM routine มากกว่าทางด้าน hardware

16. **Software package** *ต่าง ๆ ของ* **Mac** *นำมา* **Integrate** *กันได้หรือไม่?*

■ ในการพิจารณาถึงการ integrate software นั้นต้องคำนึงถึง 2 ประการด้วยกันคือ.....

ประการแรกคุณสามารถ cut และ paste software package สำหรับ Mac ที่ผลิตโดย Apple เองได้ เช่นระหว่าง Mac Paint กับ Mac Write

ประการที่สอง software ส่วนใหญ่ของ third party นั้นคือมาตรฐานของ protocal ซึ่งใช้ routine ใน ROM อยู่แล้วจึงสามารถนำมา integrate กันได้โดยไม่มีปัญหาอะไร

17. มีใครเขียนเกมส์ให้ **Macintosh** *บ้างหรือเปล่า?*

■ Macintosh ไม่ได้เป็นไมโครฯที่เน้นทางด้าน home computer แต่อย่างไรก็ตามตามรายชื่อผู้ที่พัฒนาโปรแกรมให้ Mac อยู่ในขณะนี้มีอยู่หลายบริษัทที่เป็นบริษัทผู้ผลิตเกมส์ชั้นนำ เราอาจจะได้เห็นเกมส์สำหรับ Mac ในเร็ว ๆ นี้ก็ได้

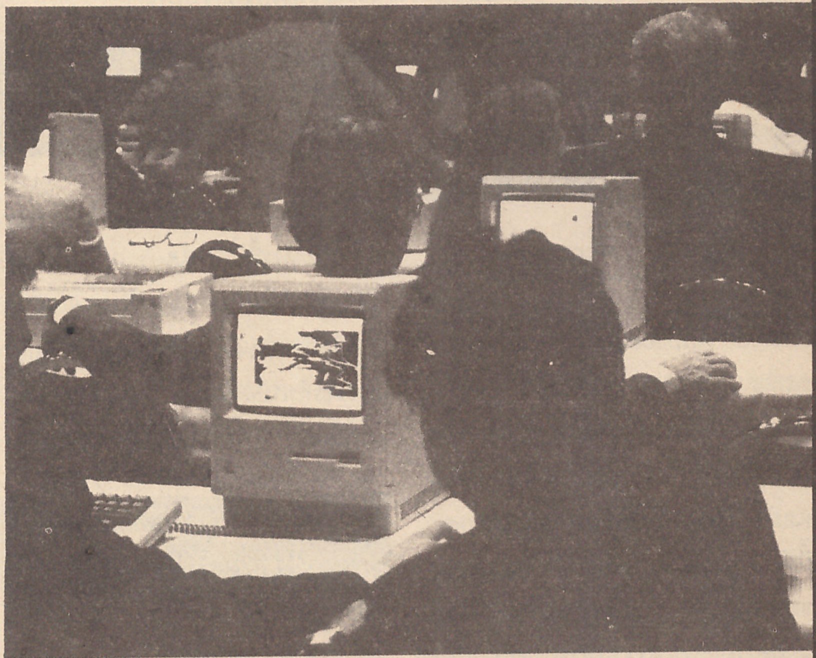
18. **Disk** *ของ* **Mac** *สามารถ* **"Write Protect"** *ได้หรือไม่*

■ Disk ของ Mac สามารถ Write Protect ได้เช่นเดียวกับ disk ประเภทอื่น ๆ

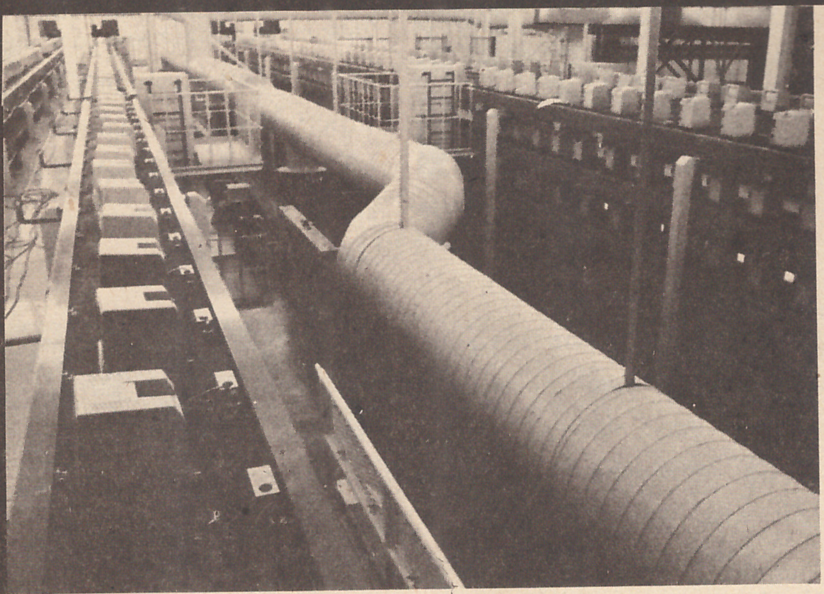
19. **Finder** *คืออะไร?*

Finder คือ software ตัวหนึ่งที่ใช้จัดการกับ file ใน system และใน disk

Finder สามารถทำให้คุณสามารถเก็บเอกสารไว้ในแฟ้มสามารถตัดเชื่อมข้อมูลจาก file หนึ่งไปยังอีก



รูปที่ 4 Dealer ของ Apple กำลังหัดใช้ Mac



รูปที่ 5 โรงงานหุ่นยนต์ มูลค่า 20 ล้านเหรียญ ของ Apple สามารถผลิต Mac ได้ 1 เครื่องทุก ๆ 27 วินาที

file หนึ่งตลอดจน function ที่ใช้บ่อย ๆ เช่นการ copy file

Finder ช่วยให้ไม่ต้องใช้คำสั่งซับซ้อนหรือชื่อต่าง ๆ มากมายในการเก็บหรือเรียกเอกสารมาใช้ทั้งนี้โดยใช้เทคนิครูปภาพและ mouse ซึ่งเป็นเทคโนโลยีของ Lisa ก็สามารถจัดการได้ เหมือนกับการทำงานอยู่บนโต๊ะในชีวิตประจำวัน

20. Desk Accessories *คืออะไร?*

■ Desk Accessory คือ function การทำงานที่เป็น function หลักของ Application program ทุกตัว อันได้แก่ นาฬิกา, เครื่องคิดเลข, สมุดโน้ต ฯลฯ ซึ่งแสดงเป็นรูปภาพแทนคำสั่งต่าง ๆ ในภาษาคอมพิวเตอร์ Desk Accessory นี้เป็น open-ended ซึ่งหมายความว่า Apple และบริษัทซอฟท์แวร์ third party ต่าง ๆ สามารถเพิ่มเติมขึ้นได้ในอนาคต

21. *จะใช้ภาษา* Pascal *กับ* Macintosh *ได้ไหม?*

■ สามารถใช้ได้แต่ Apple ไม่แนะนำ ควรใช้เป็นเพียง teching tool เท่านั้นเนื่องจากไม่เหมาะสำหรับการใช้งานโปรแกรมใหญ่ ๆ

22. FORTRAN *และ* COBOL *จะใช้ได้ไหม?*

Apple ยังไม่มีแผนการที่จะพัฒนาภาษาอื่นกับ Mac นอกจาก BASIC, PASCAL และ LOGO เนื่องจากจำเป็นต้องใช้ memory มาก อย่างไรก็ตามผู้พัฒนา third party software อาจทำให้ก็ได้

23. Macintosh compatible *กับ* Lisa *หรือไม่?*

■ ซอฟท์แวร์ที่พัฒนาสำหรับ Macintosh สามารถ run กับ Lisa ได้เนื่องจาก Lisa 2 (Lisa รุ่นใหม่) ใช้ disk $3\frac{1}{2}$ นิ้ว เหมือนกัน จึงใช้โปรแกรมร่วมกับและอ่าน data file กับได้ (ส่วน Lisa รุ่นแรกก็สามารถ upgrade ให้ใช้ disk $3\frac{1}{2}$ นิ้วได้เช่นกัน

24. *จะใช้ระบบ* Data Communication *อะไรกับ* Mac *ได้บ้าง?*

■ โดยการใช้โปรแกรม Mac-terminal กับ Apple Modem ทำให้สามารถติดต่อกับบริการข่าวสารอิเลคทรอนิคส์ต่าง ๆ หรือไมโครคอม-พิวเตอร์อื่นๆ ตลอดจนเมนเฟรมต่าง ๆ โดยการใช้ TTY, VT 100 และ VT 52 emulation ส่วนการติดต่อแบบ terminal 2370 IBM mainfram ใช้ผ่านทาง Apple Line coax adapter unit

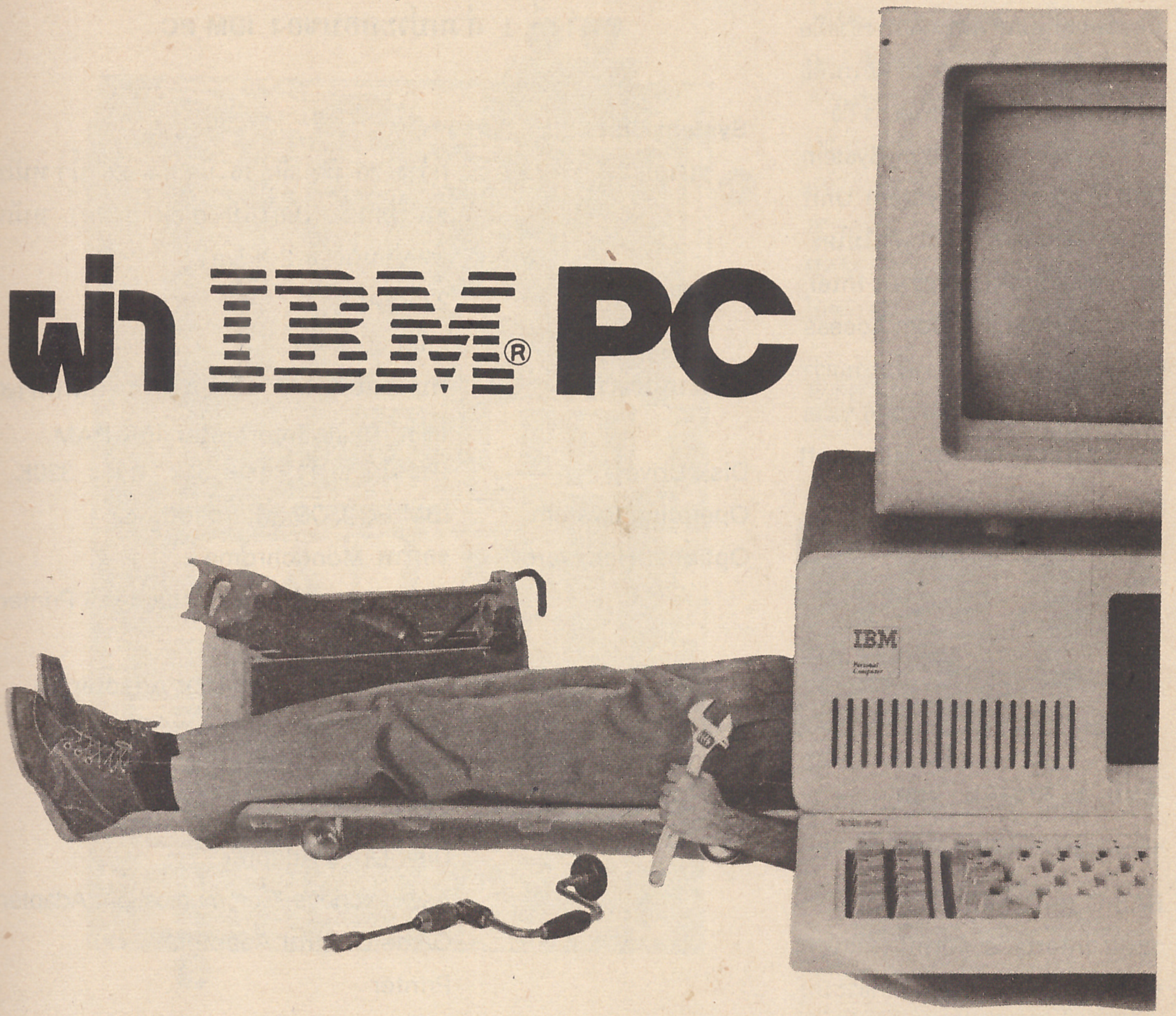
25. Mac *จะ* Compatible *กับ* Apple II *และ* Apple III *หรือไม่?*

■ เนื่องจากเทคโนโลยีและ hardware ของ Macintosh (ใช้ 68000) กับ Apple II และ Apple III (ใช้ 6502) แตกต่างกันมากจึงไม่ Compatible กัน

26. Macintosh *ใช้* Operating System *อื่นได้หรือไม่?*

■ Apple ไม่มีแผนการที่จะนำ Operating System อื่นมาใช้ เนื่องจาก OS ของ Mac นั้นออกแบบมาเป็นพิเศษหรือให้อาศัยข้อได้เปรียบของ hardware ของ Mac มากที่สุดแล้ว

ผ่า IBM® PC



ตอนที่ 1 แนะนำฮาร์ดแวร์

การที่ IBM ได้นำเครื่องคอมพิวเตอร์ส่วนบุคคลหรือที่เรียกว่า IBM PC (IBM Personal Computer) ออกสู่ตลาดนั้น ได้ก่อให้เกิดการปฏิวัติครั้งสำคัญในวงการอุตสาหกรรมของไมโครคอมพิวเตอร์ หลายคนได้ยกย่องว่า IBM PC เป็นไมโครคอมพิวเตอร์ที่ดีที่สุดเท่าที่มีการนำเสนอต่อตลาดก่อนหน้านี้นั้น ตลาดไมโครคอมพิวเตอร์มีผู้ครองตลาดเป็นผู้นำอยู่ 3 บริษัทคือ บริษัท Apple Computer ผู้ผลิต Apple, Commodore International ผู้ผลิต Commodore และบริษัท Tandy ผู้ผลิต Radio Shack

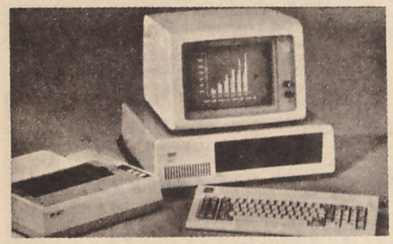
การปรากฏตัวของ IBM PC จึงก่อให้เกิดความปั่นป่วนในตลาดอย่างหลีกเลี่ยงไม่ได้ ยอดขายของ IBM PC คาดว่าจะเพิ่มขึ้นจาก 360 ล้านดอลลาร์ในปี 1982 เป็น 2.3 พันล้านดอลลาร์ในปี 1986 ส่วนซอฟท์แวร์ของ IBM จะเพิ่มจาก 85 ล้านดอลลาร์ในปี 1982 เป็น 700 ล้านดอลลาร์ในปี 1986 และอุปกรณ์ฮาร์ดแวร์ประกอบอื่น ๆ ก็จะเติบโตขึ้นจาก 65 ล้านดอลลาร์ 685 ล้านดอลลาร์ในปี 1986 และยอดขายซอฟท์แวร์ของบริษัทอื่น ๆ ก็จะเพิ่มขึ้นจาก 15 ล้านดอลลาร์ในปี 1982 เป็น 395 ล้านดอลลาร์ในปี 1986

ยอดขายทั้งหมดข้างต้นนั้นมีมูลค่ามากกว่า 4 พันล้านดอลลาร์ในปี ค.ศ. 1986 หรือพูดง่าย ๆ ก็คือ *IBM จะสามารถสร้างธุรกิจขนาด 4 พันล้านดอลลาร์ขึ้นได้โดยการออก IBM PC เพียงตัวเดียว*

ระบบหลักของ IBM PC ประกอบด้วย 2 หน่วยคือ คีย์บอร์ด และ System Unit ดังแสดงไว้ในรูปที่ 1 หน่วยคีย์บอร์ดนั้นจะประกอบด้วยคีย์บอร์ดหลักซึ่งมีลักษณะการวางตำแหน่งคีย์เหมือนกับเครื่องพิมพ์ดีดมาตรฐานทั่วไปทางขวาของคีย์บอร์ดหลักจะมี numeric keypad ซึ่งอำนวยความสะดวกในการ

ป้อนข้อมูลที่เป็นตัวเลขและทางซ้ายมือจะมีฟังก์ชั่นคีย์จำนวน 10 คีย์บนคีย์บอร์ดหลักจะมีคีย์อยู่ทั้งหมด 83 คีย์

หน่วยคีย์บอร์ดจะต่ออยู่กับ System Unit ด้วยสายเคเบิ้ล System Unit จะบรรจุหัวใจของ PC นั่นคือไมโครโปรเซสเซอร์ 8088 ของบริษัท Intel, dynamic RAM (Random Access Memory) ขนาด 64 K หรือมากกว่า ROM (Read only Memory) ขนาด 40K ซึ่งบรรจุ extended Microsoft



รูปที่ 1 SYSTEM UNIT

BASIC เอาไว้ ดิสค์ไดร้ว์ 1 หรือ 2 ตัว ระบบ interface กับเทป (คาสเซท) ลำโพงและ Slot สำหรับขยายระบบอีก 5 Slot

อุปกรณ์ประกอบอื่น ๆ ที่จะทำให้ IBM PC กลายเป็นระบบไมโครคอมพิวเตอร์ที่สมบูรณ์ ได้แก่ *จอภาพสีเดียว* ***(IBM Morochrome Display)*** *ซึ่งเป็นส่วนแสดงผลด้วยวีดีโอบางระบบและเครื่องพิมพ์แบบเมทริกซ์* ***IBM 80 CPS*** *ซึ่งสามารถพิมพ์ได้ด้วยความเร็ว 80 ตัวอักษรต่อวินาทีทั้งหมด จะแสดงใน ตารางที่ 1*

ตารางที่ 1 ส่วนประกอบของ IBM PC

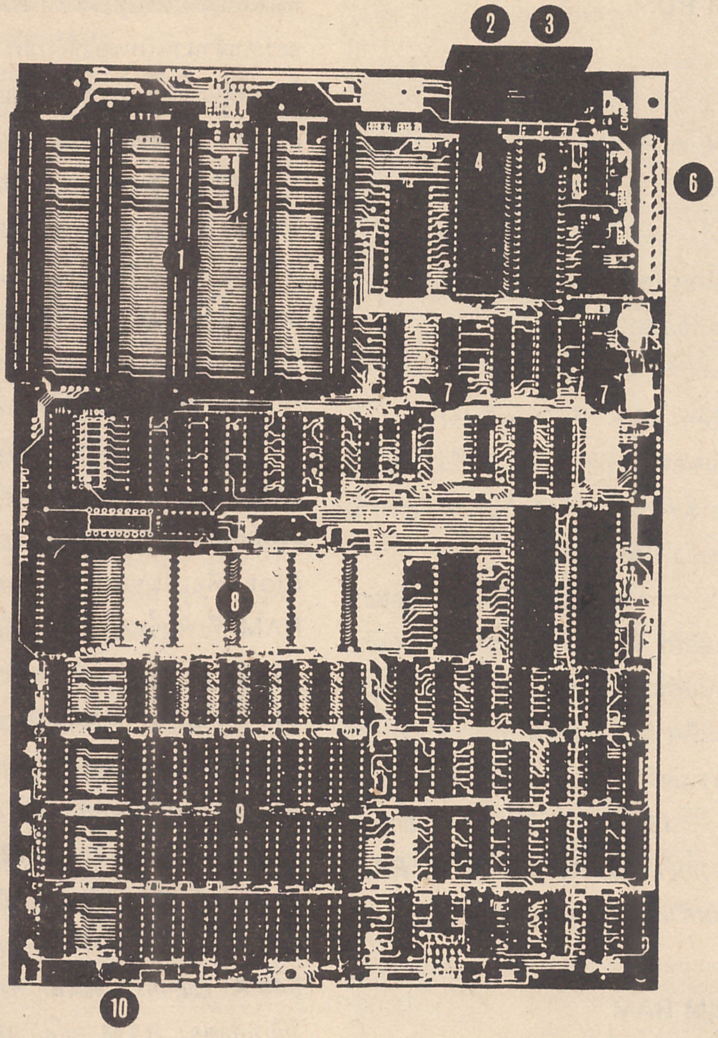
System Unit	
ขนาด	: กว้าง 20 นิ้ว ลึก 16 นิ้ว สูง 5.5 นิ้ว หนัก 21 ปอนด์ (เมื่อไม่มี disk drive) หนัก 25 ปอนด์ (มี 2 drive)
ใช้ไฟฟ้า	: 220 VAC
โปรเซสเซอร์	: Intel 8088
ระบบมาตรฐาน	: หน่วยคีย์บอร์ด และ สายเคเบิ้ลต่อคาสเซท Basic Interpreter 16K RAM
Disk Drive	บรรจุได้ 2 ตัว ขนาด 5¼" ตัวละ 320K
Operating System	IBM PC DOS (Microsoft)
Optional Hardware	จอภาพ Monochrome
	Monochrome Display Adapter 8 Printer Adapter
	Color Graphic Monitor Adapter
	16K RAM expansion
	32K RAM expansion
	64K RAM expansion
	Disk Drive Adapter
	Asynchronous Communication Adapter
	Game Control Adapter
	Printer
	Printer Adapter
	Printer Cable
	Printer Stand
Software	BASIC Interpreter (Microsoft) standard
	Extended BASIC Interpreter (Microsoft)
	PASCAL Compiler (Microsoft)
	Visicalc
	PeachtreeAccounting System
	Advanced Diagnostic Package

System Board

System Board หรือ Mother Board คือแผ่นวงจรหลักของระบบไมโครคอมพิวเตอร์วงจรส่วนใหญ่ของ PC จะอยู่บน System board รวมทั้ง ไมโครโปรเซสเซอร์, ROM, RAM, DMA, Expansion Slot, Cassette, Interface, Keyboard, Interface, Speakers & Timing Circuit

ROM และ RAM

ROM ย่อมาจาก Read Only Memory ซึ่งหมายถึงหน่วยความจำที่สามารถอ่านได้เพียงอย่างเดียว ROM



รูป 2 แสดง **IBM System Board**

จะสามารถบันทึกข้อมูลได้อย่างถาวร ถึงแม้ว่าจะปิดไฟเครื่อง PC ข้อมูลใน ROM ก็จะยังคงอยู่ ในที่นี้ ROM จะเป็นตัวเก็บ BASIC Interpreter

RAM ย่อมาจาก Random Access Memory เป็นหน่วยความจำที่เราสามารถจะอ่าน (Read) และเขียน (Write) ข้อมูลได้ แต่ไม่สามารถรักษาข้อมูลได้เมื่อปิดไฟ เมื่อข้อมูลถูกโหลดมาจากเทปคาสเซทดิสค์ไดร้ว์ หรือจากคีย์บอร์ด มันจะถูกเก็บไว้ใน RAM

Dynamic และ Static Ram

RAM มี 2 ชนิดด้วยกันคือ dynamic RAM และ Static RAM Dynamic RAM สามารถเก็บข้อมูลไว้ได้ในช่วงระยะเวลาอันสั้นเพียงไม่กี่มิลลิวินาที ดังนั้นข้อมูลใดที่จะเก็บไว้ใน dynamic RAM จะต้องมีการบันทึกหรือเขียนซ้ำอยู่ตลอดเวลาหรือที่เรียกว่า ***"refresh"*** ฟังก์ชั่นหน้าที่ที่ทำการ refresh นี้จะต้องเป็นส่วนหนึ่งของวงจรสนับสนุนเมื่อ dynamic RAM ได้รับการออกแบบมา

Static RAM นั้นมีราคาแพงกว่า dynamic RAM อย่างไรก็ตาม เมื่อข้อมูลถูกเขียนลงบน Static RAM แล้วมันจะคงอยู่ตราบเท่าที่มีไฟเลี้ยงอยู่เท่านั้น รายละเอียดของ ROM และ RAM ใน IBM PC ได้แสดงไว้แล้วในตารางที่ 2 และ 3 ตามลำดับ

IBM ROM

ดังได้กล่าวแล้วในตอนที่แล้วว่า ROM เป็นหน่วยความจำที่สามารถบันทึกข้อมูลไว้ได้อย่างถาวร IBM PC ใช้ ROM หรือที่จะบันทึกโปรแกรมที่จำเป็นในการปฏิบัติงานของคอมพิวเตอร์ โปรแกรมที่บรรจุใน ROM นี้มักจะเรียกกันว่า "firmware"(คือ software ใน hardware) ROM ใน IBM PC จะบรรจุ BIOS, BASIC Interpreter และ bootstrap loader

bootstrap loader คือโปรแกรมที่ใช้ในการสตาร์ทดิสค์ไดร้ว์ และโหลดโปรแกรมควบคุมการปฏิบัติงานหรือ Operating System จาก System diskette ROM ของ IBM PC จะอยู่ใน System Board ถึงแม้ว่าใน System board จะจุชิพ ROM ขนาด 8 K ได้ถึง 6 ตัว แต่ปัจจุบันใน socket จะมีใส่ไว้เพียง 5 ตัวเท่านั้นซึ่งมีความจุรวมกัน 40 K Socket ตัวที่ 6 คงจะมีโอกาสใช้เมื่อมีการปรับปรุงพัฒนาเครื่อง PC ในอนาคต

นอกเหนือไปจาก ROM ที่อาจเพิ่มได้จนถึง 64 K ใน System board แล้วยังอาจเพิ่มได้อีกถึง 192 K โดยใช้ ROM expansion card เสียบใน expansion slot ความจุของ ROM ที่เพิ่มขึ้นนี้อาจจะใช้บรรจุ

ตารางที่ 2 IBM ROM

ผู้ผลิต	: Motorola
ความจุสูงสุด	: 256K (48K ใน System board)
ขนาดใช้จริง	: 40K
โปรแกรมใน	: BIOS
	BASIC Interpreter
	bootstrap loader

ซอฟท์แวร์ใช้งานต่าง ๆ ลงใน ROM หรือ BIOS routines ต่าง ๆ สำหรับเพิ่มอุปกรณ์เพิ่มเติมสำหรับ PC ในอนาคต

IBM RAM

RAM ของ IBM มีบรรจุมาใน System board 16 K RAM อีก 48 K อาจจะนำมาเพิ่มใน System board ได้ 16 K แรกของ RAM จะบัดกรีติดกับ System board เลยในขณะที่อีก 48 K ที่เพิ่มมานั้นจะเสียบอยู่ในซอคเก็ททำให้ผู้ใช้แม้จะไม่มีความรู้ทางเทคนิคมากนักก็สามารถขยายความจุของ RAM ได้เอง

ทุก 16K ของ RAM บน System baard จะประกอบด้วยชิพ 9 ชิพด้วยกันแต่ละตัวมีความจุ 2K หลายคนอาจจะคิดว่านำจะมีชิพเกินมา 1 ชิพ แต่ความจริงไม่เท่า Parity check ***IBM PC** ใช้ชิพตัวที่ 9 สำหรับทำหน้าที่ **parity check** กับ **RAM** ชิพตัวอื่น ๆ* Parity check คือการทดสอบที่ทำตลอดเวลาใน RAM โดยจะบอกจำนวนบิทของข้อมูลที่ "ON" ข้อมูลนี้จะใช้ในการตรวจสอบในการส่งและรับข้อมูลเพื่อให้ถูกต้องตรงกัน ถ้า RAM ไม่ผ่าน parity check ข้อความต่อไปนี้จะปรากฏบนจอภาพ :

PARITY ERROR

และจะไม่สามารถ RUN โปรแกรมที่กำลังใช้อยู่ได้

นอกเหนือจาก RAM ขนาด 64 K ที่สามารถบรรจุลงไปใน System board แล้วยังสามารถใช้ RAM Expansion Card เสียบลงใน Expansion Slot เพื่อเพิ่มขนาดของ RAM ได้ RAM ที่เพิ่มขึ้นนี้สามารถเพิ่มขึ้นได้ทีละ 32 K หรือ 64 K จนกระทั่งถึง 544 K *ไมโครโปรเซสเซอร์ 8088 ของ **PC** สามารถจะแอดเดรสข้อมูลในหน่วยความจำได้ถึง 1 เมกกะไบท์ (1 ล้านไบท์) ในหนึ่งเมกกะไบท์นั้น **128 K** จะถูกใช้สำหรับ **Graphic and Display Adapter, 256 K** ใช้สำหรับ **ROM 640 K** ใช้สำหรับ **RAM** ดังนั้นจึงยังมีที่เหลือของ **RAM** อยู่อีก **384 K** ซึ่งคงจะมีการพัฒนาใช้งานต่อไปในอนาคต*

ตารางที่ 3 IBM RAM

ชนิด	ชนิด 4116 ขนาดตัวละ 2K
	Access time 250 nanosecond
	Cycle time 450
การจัดเรียง	แถวละ 16K แถวละ 9 ชิพ
	(ตัวที่ 9 ใช้สำหรับ parity check)
	แถวที่ 1 บัดกรีกับแผงวงจร ที่เหลือเสียบใน socket
	32K Expansion Board มี 2 แถว
	64K Expansion Board มี 4 แถว
ขนาดความจุ	มาตรฐาน 16K
	ขยายใน System board อีก 48K
	ขยายด้วย Expansion Slot อีก 192K
	อาจจะขยายได้ในอนาคตอีก 384K
ชนิด	Dynamic RAM ใช้ได้กับ DMA Direct Memory Addressing)

IBM BIOS

BIOS** ย่อมาจาก**BASIC Input /Output System Bios เป็นกลุ่มของโปรแกรมซึ่งควบคุมการส่งผ่านข้อมูลระหว่างไมโครโปรเซสเซอร์และอุปกรณ์ต่าง ๆ ที่มาเชื่อมต่อกับมัน อาทิเช่น เครื่องพิมพ์ จอวีดีโอ คีย์บอร์ด และ Modem ฯลฯ

BIOS ของ PC จะประกอบด้วย

- ***Cassette Tape Operating System***
- ***Power on self Test***
- ***Graphic character Generator***

- ***System Configuration analysis Program***
- ***Time of Day Clock***
- ***Mini Floppy Disk Drive***
- ***Bootstrap Loader***

ไมโครคอมพิวเตอร์ที่ออกแบบมาให้เป็น 8 บิท จะถูกจำกัดทั้งจำนวนและขนาดของโปรแกรมที่บรรจุลงใน BIOS ทั้งนี้เพราะว่าไมโครโปรเซสเซอร์ 8 บิท สามารถแอดเดรสหน่วยความจำได้เพียงเศษส่วนอันน้อยนิดของโปรเซสเซอร์ 16 บิท IBM PC เป็นไมโคร คอมพิวเตอร์ขนาด 16 บิท ซึ่งสามารถแอดเดรสหน่วยความจำได้ถึง 1 เมกกะไบท์ ทำให้สามารถใช้งาน ROM ได้อย่างกว้างขวาง

*ข้อดีอย่างหนึ่งของการบรรจุ **BIOS** ไว้ใน **ROM** คือโปรแกรมจะสามารถเรียกใช้ได้ทันทีที่เปิดเครื่อง แต่อย่างไรก็ตามก็มีข้อเสียก็คือ **BIOS routines** ต่าง ๆ ไม่สามารถเปลี่ยนแปลงได้ เช่น เมื่อต้องการเพิ่มอุปกรณ์ใหม่ ๆ เข้ามา*

IBM CPU

CPU (Central Processing Unit) หรือหน่วยประมวลผลกลางเป็นหัวใจของคอมพิวเตอร์ทุกเครื่อง CPU จะเป็นตัวควบคุมการทำงานของอุปกรณ์ต่าง ๆ ของคอมพิวเตอร์

ในคอมพิวเตอร์เครื่องใหญ่ CPU และ ALU (Arithmetic logic unit) ประกอบด้วยกลุ่มของชิพไอซีที่แบ่งแยกกันทำงาน ในคอมพิวเตอร์เครื่องเล็ก CPU และ ALU จะอยู่รวมกันในชิพเดียวกันเรียกว่า "ไมโครโปรเซสเซอร์"

*ตัวไมโครโปรเซสเซอร์นั้นสามารถให้คำจำกัดความได้ว่าเป็นชิพไอซีชิพเดียวที่ประกอบด้วยวงจรลอจิกของหน่วยประมวลผลกลางและวงจรลอจิกอื่น ๆ ที่ใช้ประกอบกับ **CPU***

ไมโครโปรเซสเซอร์ที่ใช้ใน PC คือ ***Intel 8088*** ซึ่งผลิตโดยบริษัท Intel Corporation แห่ง Sunnyvale California Intel เป็นผู้บุกเบิกในการพัฒนาทางด้านไมโครโปรเซสเซอร์ และได้พัฒนาต้นตระกูลของ 8088 นั่นคือ ชิพคำนวณ 4004 ขึ้นในราวต้นทศวรรษ 1970 ต่อมา Intel ก็ได้พัฒนา 8088, 8080, 8085 และ 8088/8086 ขึ้นมา ความแตกต่างระหว่าง 8088 และ 8086 เชิญอ่านได้ในฉบับนี้

ไมโครโปรเซสเซอร์ลอจิกนั้นอยู่บนพื้นฐานของหน่วยนี้เรียกว่า "บิท" บิท คือพื้นฐานของการเก็บระบบข้อมูลภายในคอมพิวเตอร์ หนึ่งบิทจะประกอบด้วยสถานะของสวิทช์ 2 สถานะคือ เปิด (ON) หรือ ปิด (OFF)

บิทมักจะถูกแยกเป็นกลุ่มที่ละ 8 ซึ่งเรียกว่า "ไบท์" หนึ่งไบท์จะสามารถแทนตัวอักษรได้หนึ่งตัว (หรือเป็นตัวเลขหรือสัญญลักษณ์ก็ได้) โดยทั่วไปแล้วคอมพิวเตอร์จะจัดการกับข้อมูลทีละ 2 ไบท์

ไมโครโปรเซสเซอร์ 8 บิท ส่วนมากสามารถแอดเดรสข้อมูลในหน่วยความจำได้ 65,535 ไบท์ หรือ 64K ถึงแม้ว่ามันดูจะเหมือนว่ามากแต่มันก็สามารถจุข้อมูลขนาดเอกสารประมาณ 30 หน้า ได้เท่านั้น

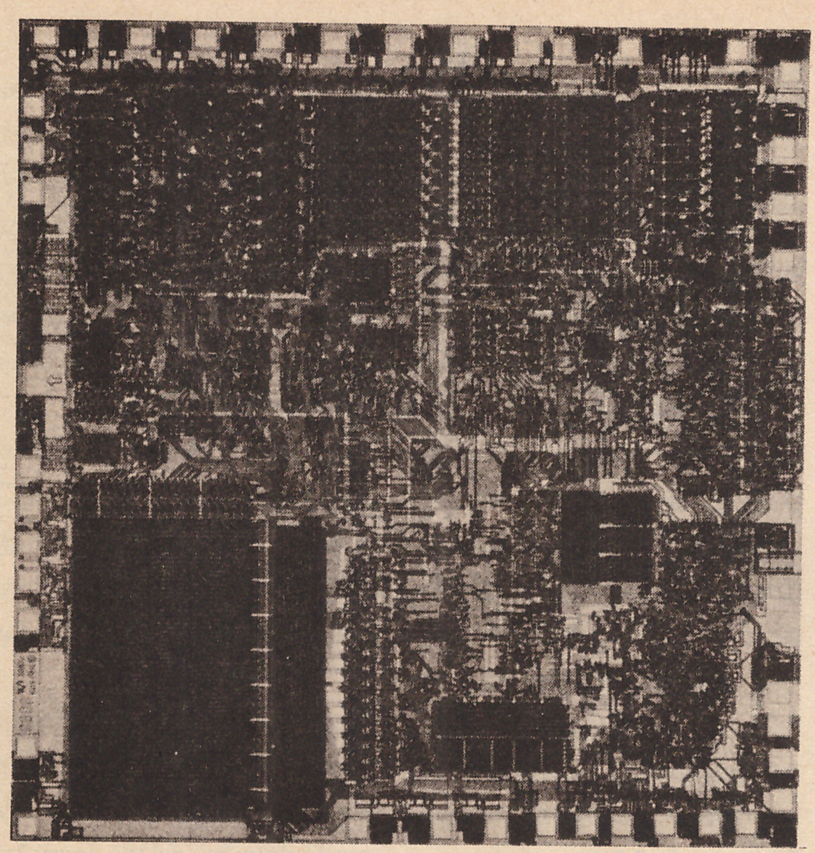
ไมโครโปรเซสเซอร์ 16 บิท ส่วนมากสามารถแอดเดรสตั้งแต่ 256,000 ถึง 16 ล้านไบท์ของหน่วยความจำ ยิ่งไปกว่านั้นไมโครโปรเซสเซอร์ 16 บิท สามารถจัดการกับข้อมูลด้วยความเร็วมากกว่าไมโครโปรเซสเซอร์ 8 บิท 2 ถึง 10 เท่าตัวทีเดียว

Intel 8088

ข้อได้เปรียบสำคัญประการหนึ่งของ IBM PC ที่มีเหนือกว่าไมโครคอมพิวเตอร์ทั่ว ๆ ไปก็คือ *ประสิทธิภาพ*

ตารางที่ 4 IBM BIOS

วัตถุประสงค์	: ควบคุมการส่งผ่านข้อมูลระหว่างไมโครโปรเซสเซอร์และอุปกรณ์เชื่อมต่อต่างๆ
ออกแบบโดย	: Microsoft Inc
ตำแหน่ง	: 8K ROM ชิพใน System Board
โปรแกรม	: 1. Cassette Operating System 2. Power on Self-test 3. I/0 routines สำหรับ video display Keyboard printer communication adapter 4. Graphic Character Generator 5. System Configuration Analysis 6. Time-of-Day Clock 7. Mini Floppy disk Drive Bootstrap Loader



รูปที่ 3 ชิพท์ของ Intel 8088

การทำงานซึ่งได้มาจากการใช้ ***CPU 8088***

8088 สามารถแอดเดรสข้อมูลได้ถึง 1 ล้านไบท์ (1 Megabyte ย่อว่า 1 MB) 8088 มี address line 20 เส้น ซึ่งทำให้มี 2 ยกกำลัง 20 combination (2^{20}) ซึ่งก็เท่ากับ 1,048,576 แอดเดรสต่าง ๆ กับที่มันจะเข้าถึงได้

ตารางที่ 5 คุณสมบัติของ Intel 8088

โครงสร้าง	: HMOS บรรจุใน package พลาสติก 40 ขา
ผู้ผลิต	: Intel Corporation
ออกสู่ตลาด	: 1979
คุณสมบัติ	: 1. มีสายแอดเดรส 20 เส้น 2. แอดเดรสหน่วยความจำได้ 1 MB 3. ทำงานได้ 650,000 คำสั่งต่อวินาที 4. มีสายข้อมูล 8 เส้น 5. ใช้นาฬิกาความถี่ 4.77 MHZ 6. bus interface และ execution unit อยู่แยกจากกัน 7. คำสั่งภาษาเครื่องพื้นฐาน 99 คำสั่ง 8. ซอฟท์แวร์ใช้ได้กับ Intel 8086

ความเร็วของ 8088 คือประมาณ 0.65 mps (1 ล้านคำสั่งต่อวินาที) หรืออีกนัยหนึ่ง 8088 จะสามารถทำการคำนวณคณิตศาสตร์ หรือส่งผ่านข้อมูลได้ 650,000 ครั้งต่อวินาที ซึ่งเมื่อเทียบกับ 8080 8088 จะเร็วกว่าประมาณ 6 เท่า

คุณสมบัติของ 8088 จะแสดงไว้ในตารางที่ 5

Intel 8087 Numeric Data Processor

CPU ส่วนใหญ่จะทำการคำนวณทางคณิตศาสตร์ได้ค่อนข้างช้า โดยเฉพาะอย่างยิ่งเมื่อเป็นการคำนวณที่ไม่ใช่จำนวนเต็ม

เพื่อเป็นการเพิ่มความเร็วในการคำนวณทางคณิตศาสตร์ คอมพิวเตอร์ มินิ และเมนเฟรมส่วนใหญ่จะใช้โปรเซสเซอร์ต่างหากในการคำนวณตัวเลขซึ่งเรียกมีชื่อเรียกว่า Math processor หรือ floating point co-processor

โปรเซสเซอร์คณิตศาสตร์จะประกอบด้วยวงจรลอจิกภายในที่จำเป็นที่ต้องใช้ในการคำนวณตัวเลขเท่านั้น โดยไม่เกี่ยวข้องกับคำสั่งโปรแกรมภายนอก โปรเซสเซอร์คณิตศาสตร์จะสามารถทำงานได้เร็วกว่าธรรมดาอาจถึง 100 เท่า

Intel 8087 เป็น Numeric Data Processor ซึ่งทำหน้าที่เป็นโปรเซสเซอร์ร่วมของ CPU 8086 และ 8088

ใน ***System board*** *ของ* ***IBM PC*** *จะมี* ***Socket*** *ขนาด 40 ขาอยู่ชุดหนึ่งมีชื่อว่า* ***AUX Processor Socket*** ที่นี้เองเป็นที่ที่เตรียมไว้ให้สำหรับโปร

ตารางที่ 6 คุณสมบัติของ Intel 8087

ผู้ผลิต	:	Intel Corporation
ออกสู่ตลาด	:	1980
คุณสมบัติ	:	มีฟังก์ชั่นคณิตศาสตร์อยู่ภายในตัว เช่น บวก ลบ คูณ หาร รากที่ 2 ค่าสัมบูรณ์ tangent arctangent ฯลฯ 2. จัดการตัวเลขจำนวนเต็มได้ 18 หลัก และทศนิยมได้กว้างตั้งแต่ 16 ถึง 80 บิท 3. ใช้งานได้กับทั้ง 8086 และ 8088

เซสเซอร์ร่วม ***IBM PC 8087*** *จะทำให้การคำนวณตัวเลขของ* ***IBM PC*** *เร็วขึ้นอย่างน้อย 20 เท่า*

คุณสมบัติของ Intel 8087 แสดงไว้แล้วในตารางที่ 6

Microsoft BASIC Inter - preter

Microsoft BASIC คือภาษา BASIC มาตรฐานของ IBM PC BASIC Interpreter มีขนาด 40 K บรรจุอยู่ใน ROM รูปแบบต่าง ๆ ของ Microsoft BASIC จะอธิบายอย่างละเอียดในตอนต่อ ๆ ไป สำหรับตอนนี้ถือว่าเป็นเพียงการแนะนำเท่านั้น

IBM PC ใช้ภาษา BASIC ที่แตกต่างกันถึง 3 แบบคือ ***Casette BASIC, DISK BASIC*** *และ* ***Advanced BASIC***

Cassette BASIC *คือภาษา* ***BASIC*** *ที่ใช้เมื่อใช้เทปคาสเซทในการเก็บข้อมูล* ***Cassette BASIC*** *ไม่สามารถใช้ได้ถ้าข้อมูลถูกบันทึกอยู่ในดิสเก็ท*

Disk BASIC *มีฟังก์ชั่นการทำงานทั้งหมดของ* ***Cassette BASIC*** *และยังสามารถใช้ได้เมื่อข้อมูลถูกบันทึกอยู่บนแผ่นดิสเก็ทหรืออาจจะกล่าวได้ว่า* ***Disk BASIC*** *ใช้ได้กับข้อมูลที่อยู่บนดิสเก็ทหรือบนเทปคาสเซท*

Advanced BASIC *มีฟังก์ชั่นการทำงานทั้งของ* ***Disk BASIC*** *และ* ***Cassette Basic*** *และยังมีฟังก์ชั่นอื่น ๆ นอกเหนือไปจากนี้อีก เมื่อมีคำสั่งด้านกราฟิคเช่น* ***CIRCLE, DRAW*** *และ* ***PAINT*** *ทำให้ผู้ใช้สามารถใช้ความสามารถทางด้านกราฟิคของ* ***IBM PC*** *ได้อย่างเต็มที่โดยใช้ร่วมกับ* ***Color/Graphic Monitor Adapter*** *และจอมอนิเตอร์สีหรือของทีวีสี*

ทั้ง Advanced BASIC และ Disk BASIC ต้องการ DOS ก่อนที่มันจะทำงานได้หรือก่อนที่โปรแกรมใดๆ ที่เขียนด้วยโปรแกรมทั้ง 2 นี้จะทำงานได้ DOS จำเป็นสำหรับการจัดการ input และ output จาก disk drive ของคุณ

IBM PC Peripherals

ในตอนท้ายของตอนที่ 1 เราจะศึกษากันถึงอุปกรณ์ประกอบต่าง ๆ ที่คุณจะสามารถเพิ่มเติมเข้าไปยังเครื่อง IBM PC ของคุณ อุปกรณ์เหล่านี้ได้แก่

- ***Disk Drive***
- ***Monochrome Display***
- ***IBM 80 CPS Printer***
- ***Monochrome Display/Printer Adapter Card***
- ***Color/Graphic Montor Adapter Card***
- ***Color Monitor***
- ***Asynchronous Commumication Adapter***

Disk Drive และ DOS

Disk Drive ของ IBM PC เป็นชนิด soft sector, double side double density drive ตัว floppy disk drive ใช้ 40 track ต่อดิสค์ และ 512 ไบท์ต่อ sector และ 8 sector ต่อ track ทำให้ drive หนึ่งจุได้ 327,680 ไบท์ หรือ 320 K

IBM Disk Operating System เป็นกลุ่มของโปรแกรมที่ทำให้ผู้ใช้จัดการกับข้อมูลต่าง ๆ ในการส่งผ่านส่วนต่าง ๆ ของคอมพิวเตอร์ เช่น diskette drive memory และจอวิดีโอ

Disk drive คืออะไร

Disk Drive เป็นส่วนประกอบที่สำคัญที่สุดส่วนหนึ่งของระบบคอมพิวเตอร์ Disk Drive สามารถเก็บข้อมูลได้เป็นจำนวนมาก ***Disk*** *ของ* ***IBM*** *ขนาด 5 1/4 นิ้ว สามารถจุข้อมูลได้ 327,680 ตัวอักษร*

Disk จะต่างจาก RAM ตรงที่ว่า เมื่อข้อมูลถูกบันทึกใน disk แล้ว มันจะไม่สูญหายไปเมื่อปิดเครื่องคอม

ตารางที่ 7

	IBM Disk Drive
ผู้ผลิต	: Tandon Magnetic Inc.
ความจุ	: Double Side/double density (DSDD)
	Unformatted 500K
	Formatted 320K (เมื่อใช้ PC DOS)
	40 track 48 track ต่อนิ้ว
	8 sector ต่อ track
	512 byte ต่อ sector
ความเร็ว	: 8ms seek time (track ถึง track)
	25 ms head setting time
	500 ms maximum start/stop time
	250 bit/second transter rate

พิวเตอร์ หรือกล่าวอีกอย่างหนึ่ง disk เป็นวิธีการบันทึกข้อมูลได้อย่างถาวร

Disk แผ่นหนึ่งจะบันทึกข้อมูลในรูปของสนามแม่เหล็กเช่นเดียวกันในเทปบันทึกเสียงทั่ว ๆ ไป ข้อแตกต่างสำคัญระหว่างเทปกับดิสค์ก็คือ สารแม่เหล็กของ disk จะฉาบอยู่บนแผ่นกลมคล้ายกับแผ่นจานเสียง และข้อมูลก็จะถูกบันทึกลงเป็นวง ๆ ซ้อนกัน แทนที่จะบันทึกตามระยะยาวอย่างในเทป

Disk Drive จะมีอุปกรณ์ที่เรียกว่า หัวอ่าน/บันทึก (Read/Write head) ซึ่งใช้ในการอ่านและบันทึกข้อมูล คอมพิวเตอร์สามารถเคลื่อนหัวนี้ไปยังตำแหน่งใด ๆ ก็ได้บนแผ่นดิสค์ ซึ่งแตกต่างจากเทปแม่เหล็กซึ่งข้อมูลจะถูกอ่านหรือบันทึกเรียงกันไป

ความสามารถในการอ่านและบันทึกข้อมูลในตำแหน่งใด ๆ ก็ได้นี้เรียกกันว่า ***Random Access*** *ส่วนในการอ่านหรือบันทึกข้อมูลเรียงกันไปตามลำดับนั้นมีชื่อเรียกว่า* ***sequential Access*** *เช่น ในเทปคาสเซท เป็นต้น*

ชนิดของ Disk

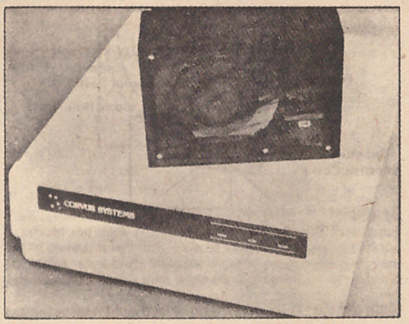
Disk ที่ใช้ในไมโครคอมพิวเตอร์มี 3 ชนิดด้วยกันคือ hard disk, Winchester disk และ floppy disk

Hard disk

Hard disk ของไมโครคอมพิวเตอร์จะสามารถจุข้อมูลได้ตั้งแต่ 5 ถึง 10 เมกกะไบท์ 1 เมกกะไบท์เท่ากับ 1 ล้านไบท์ ตัว Hard disk เองทำด้วยจานวัสดุแข็งที่เคลื่อนสารแม่เหล็กเอาไว้ Disk drive กับ Hard disk ปกติจะอยู่แยกกันในเครื่อง ทั้ง Disk drive และ Hard disk สามารถถอดเปลี่ยนได้

Winchester Disk Drive

Winchester disk drive ออกแบบมาได้จุข้อมูลมากกว่า floppy diskette มาตรฐาน 6-10 เท่า Winchester disk จะต้องรักษาไว้ให้สะอาดที่สุดเพราะมันจะเป็นอันตรายถ้าถูกฝุ่น ผง และควัน



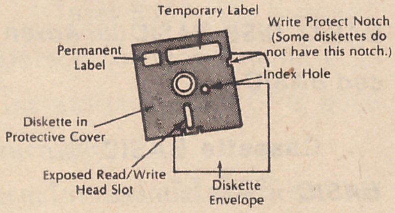
รูปที่ 4 Winchester Disk System

เนื่องจากต้องเก็บรักษาไว้ให้อ่าน Winchester disk จึงต้องซีลภายใน disk drive นั่นหมายความว่าเราไม่สามารถเปลี่ยน Winchester disk ได้

เนื่องจาก Winchester disk ไม่สามารถเปลี่ยนได้เรามักจะใช้ floppy disk ร่วมกับ Winchester disk ในการสำรองข้อมูล (Back-up) Winchester disk มักจะได้รับความนิยมในการใช้งานมากกว่า hard disk

Floppy Diskettes

Floppy Disk เป็นหน่วยเก็บข้อมูลสำรองของไมโครคอมพิวเตอร์ที่มีใช้กันมากที่สุด Floppy diskette ประกอบด้วยแผ่นไวนิลกลม ๆ สอดอยู่ในแผ่นของพลาสติค ซองนี้จะป้องกัน diskette มิได้เสียหาย ขณะใช้งาน และไม่ควรจะแกะซองออกมาเป็นอันขาด



รูปที่ 5 Floppy Diskette

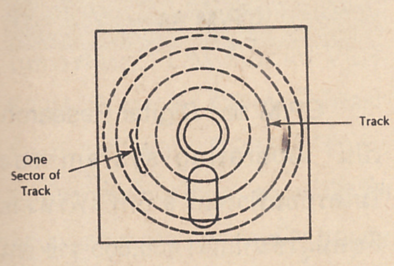
แผ่น Diskette จะหมุนอยู่ในซองป้องกันรูกลมตรงกลางทำให้ยึด diskette และหมุนมันได้ ส่วนช่องเปิดรูปรีบนซองนั้นเป็นพื้นที่ที่หัวอ่านจะเคลื่อนไปมาเวลาอ่านหรือบันทึกข้อมูล

Floppy diskette มี 2 ขนาดด้วยกันคือ 8 นิ้ว และ 5 1/4 นิ้ว ขนาด 5 1/4 นิ้ว ซึ่ง IBM PC ใช้อยู่มีชื่อเรียกว่า mini floppy diskette

Track และ Sector

เพื่อช่วยให้การค้นหาข้อมูลบน diskette พื้นผิวของ diskette จะแบ่งออกเป็น track และ sector

track จะมีลักษณะเป็นวงกลมหลาย ๆ วงซ้อนกันอยู่บนผิวของ diskette ดังแสดงในรูป 6 IBM DOS แบ่ง diskette ออกเป็น 40 track



รูปที่ 6 Tracks & Sectors

นอกจากนั้นเพื่อย่นระยะเวลาในการหาข้อมูลขึ้นไปอีก IBM DOS จึงได้แบ่งแต่ละ track ออกเป็น 8 sector ซึ่งแสดงไว้ในรูปที่ 6 เช่นกัน

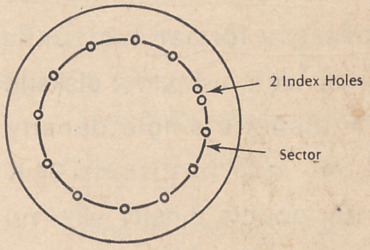
แต่ละ Sector จะจุข้อมูลได้ 512 ไบท์ เมื่อ Dos รู้ track และ sector ที่ข้อมูลนั้นอยู่ มันจะค้นข้อมูลเพียง 512 ไบท์เท่านั้นในการที่จะพบข้อมูล ๆ หนึ่ง จะเห็นได้ว่าการแบ่งพื้นผิวของ diskette ออกเป็น track และ secter จะทำให้ระยะเวลาในการหาข้อมูล (Access tine) ลดลงเป็นอย่างมาก

Hard และ Soft Sector

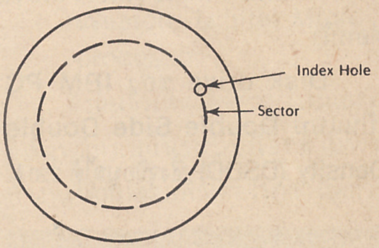
การหาตำแหน่ง track แต่ละ track บนผิวหน้าของ disk นั้นไม่ใช่ของยากตัว drive เพียงแต่เคลื่อนหัวอ่านบันทึกไปยังตำแหน่งที่ track นั้น ๆ อยู่ เหมือนกับเคลื่อนหัวเข็มของเครื่องเล่นจานเสียงไปยังเพลงใดเพลงหนึ่งบนแผ่นนั่นเอง อย่างไรก็ตามการหาตำแหน่ง Sector ใด Sector หนึ่งนั้นจะยุ่งยากขึ้นมาอีกขั้นหนึ่ง มีวิธี 2 วิธีในการหาตำแหน่ง Sector บน disk นั้นคือ hard Sectoring และ Soft Sectoring ทั้งวิธี hard Sectoring และ Soft Sectoring ต่างก็ใช้ index hole ซึ่ง index hole จะเห็นได้ในรูปที่ 5 มันอยู่ทางขวาของรูตรงกลางแผ่นดิสค์

Index hole นี้เห็นในรูปนั้นเป็นเพียงรู บน ซองพลาสติคป้องกันเท่านั้น index nole อีกอันหนึ่งจะเจาะไว้บนแผ่น diskette จริง ๆ ภายในซอง เมื่อ diskette หมุน index hole บนแผ่น diskette ก็จะเคลื่อนที่ผ่านรูบนซองป้องกัน แสงไฟภายในตัว disk drive จะส่องผ่านรูบนซอง diskette นี้เมื่อ Index hole มาอยู่ตรงรูบนซองแสงจะสามารถส่องผ่านทะลุไปยังเซ็นเซอร์ได้ ตัวเซ็นเซอร์จะส่งข่าวสารบอกตำแหน่ง index hole ไปให้ disk drive ซึ่งจะสามารถใช้ในการคำนวณตำแหน่ง Sector ต่าง ๆ ได้

ตอนนี้เราจะมาทำความเข้าใจกับความแตกต่างของ hard และ Soft sector กัน diskette ที่เป็น hard sector จะมีรูเจาะไว้ตรงตำแหน่งทุก ๆ sector รวมทั้งมีรูพิเศษอีกรูหนึ่งเพื่อใช้บอก sector แรก ตำแหน่งของ sector ต่าง ๆ ใช้วิธีนับเอาจากตำแหน่งของ sector แรก hard sectored diskette ได้แสดงไว้ในรูปที่ 7



รูปที่ 7 Hard Sectored Diskette



รูปที่ 8 Soft Sectored Diskette

Soft Sectored diskettes คือ diskette ที่มี index hole เพียงรูเดียวเจาะไว้ตรงตำแหน่งของ sector แรก โดยการคำนวณความเร็วในการหมุนของ floppy diskette (ซึ่งคงที่) ตำแหน่งที่ตั้งของ sector อื่น ๆ ก็จะสามารถคำนวณหาได้ ***IBM PC ใช้ diskette แบบ Soft Sector***

disk หน้าเดียวและ 2 หน้า

diskette บางชนิดสามารถใช้งานได้คืออ่านหรือบันทึกได้เพียงหน้าเดียวเราเรียก diskette พวกนี้จะเป็น single sided(ss)

ส่วนอีกประเภทหนึ่งออกแบบ

มาให้ใช้งานได้ทั้ง ด้านเรียกว่า Double sided (DS)

Single, Double และ Quad Diskette

Density หรือความหนาแน่น จะขึ้นอยู่กับ format ในการบันทึก ซึ่งจะนำไปสู่ความจุข้อมูลของ diskette ด้วย disgette single density 5 1/4" จะมีความจุประมาณ 94 K ถ้าเป็น double dinsity หรือความจุ 2 เท่าจะจุข้อมูลได้ราว 140-160 K และถ้าเป็น Quad density หรือความหนาแน่น 4 เท่าจะจุได้ราว 370 K ต่อหน้า

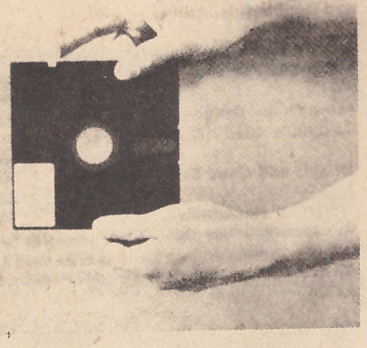
Disk drive ของ IBM PC เป็นแบบ Double Side Double Density (DSDD) ซึ่งจุข้อมูลได้ 320 K

Diskette Write Protection

ช่องบาก (notch) บนขอบด้านหนึ่งของซองป้องกันนั้นจะเป็นตัวกำหนดว่าข้อมูลจะถูกบันทึกลงบน diskette ได้หรือไม่ ใน diskette 8 นิ้วช่องบากนี้จะเรียกว่า Write protect notch ส่วนบน diskette 5 1/4" จะเรียกว่า Write enable notch ซึ่งจะมีการใช้งานตรงกันข้ามกัน

ใน diskette 5 นิ้วนั้น ข้อมูลจะไม่สามารถบันทึกลงแผ่นได้ถ้า notch นี้ไม่เปิดไว้บน diskettes ขนาด 5 1/4" ข้อมูลจะไม่สามารถบันทึกลงบนแผ่นได้ถ้า notch นี้ปิดอยู่

diskette 5 1/4" บางอันเช่น System Diskette จะไม่มี notch เลย หรือให้แน่ใจว่าไม่มีการบันทึก



รูปที่ 9 Diskette Write Protection

ข้อมูลใด ๆ ลงไปได้ คือให้อ่านได้เท่านั้น ส่วน diskette 5 1/4" ธรรมดาก็จะสามารถป้องกันการอัดเทปโดยใช้เทปปิดตรงช่อง notch ดังในรูปที่ 9

Disk Operating System

IBM PC สามารถใช้ Operating System ได้ถึง 3 ระบบคือ PC DOS, UCSD P System และ CP/M 86 ระบบทั้ง 3 นี้มีฟังก์ชั่นการทำงานแตกต่างกันทำให้คุณลักษณะของ IBM PC ขณะใช้งานแตกต่างกันไปด้วย

Operating System พอจะให้คำนิยามได้ว่าเป็นกลุ่มของ Program ซึ่งทำหน้าที่จัดการการทำงานต่าง ๆ ของคอมพิวเตอร์ หรือว่ากลุ่มของโปรแกรมที่ควบคุมการส่งผ่านข้อมูลเข้าออกระหว่างหน่วยเก็บข้อมูล เช่น disk หรือเทป

ในตอนนี้เราจะพูดถึง Operating System ของ PC โดยย่อ ๆ ส่วนรายละเอียดนั้นจะพูดถึงในตอนต่อ ๆ ไป

IBM DOS

IBM DOS เขียนขึ้นโดย Seattle Computer Product ในปี 1980 ต่อมา Microsoft Inc. ได้สิทธิ์ในการจำหน่ายระบบนี้และนำมาตั้งชื่อใหม่ว่า **MS DOS** ต่อมา Lifeboat Associates ก็ได้จัดจำหน่าย MS DOS เหมือนกันในชื่อว่า **SB - 86** ส่วน IBM นั้นใช้ MS DOS ในชื่อของ **PC DOS**

ดังนั้น PC DOS, MS DOS และ SB - 86 คือ Operating System อันเดียวกันนั่นเองในบทความนี้ เราจะเรียกระบบนี้ว่า PC DOS ตลอดไป

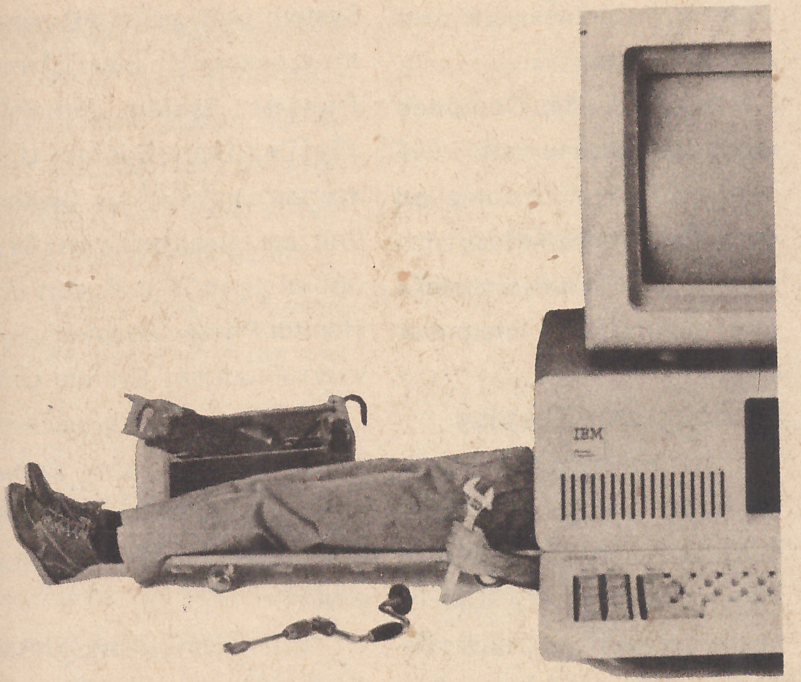
บางส่วนของ PC DOS มีความคล้ายคลึงกัน CP/M ซึ่งเป็น Operating system ซึ่งผลิตโดย Digital Research เป็นต้นว่า PC DOS ใช้เครื่องหมาย System prompt (A>) เหมือนกันกับ CP/M

อย่างไรก็ตาม PC DOS ก็ยังเป็น version หนึ่งของ Xenix operating system และ Xenix ก็ยังเป็น version หนึ่งของ Unix operating system ซึ่งพัฒนาขึ้นโดย Western Electric

CP/M 86

CP/M ของ Digital Research เป็น operating system ของไมโครคอมพิวเตอร์ 8 บิทที่แพร่หลายที่สุดมีผู้ใช้มากกว่า 400,000 ราย จนกระทั่งมีการยอมรับเป็นมาตรฐานสำหรับ 8 บิท แต่ CP/M 86 จะสามารถเป็นมาตรฐานของ 16 บิทได้หรือไม่นั้นยังเป็นที่สงสัยกันอยู่ CP/M 80 (ใช้กับ 8 bit) ปัจจุบันใช้อยู่กับเครื่องไมโครคอมพิวเตอร์ Zenith, Intertec, DEC, WANG, Xerox และ Lanier รวมทั้งดัดแปลงให้ใช้ได้กับเครื่องของบริษัท Apple และ Tandy ซึ่งเป็นผู้นำในอุตสาหกรรมนี้อยู่

CP/M พัฒนาขึ้นมาโดย ดร.แกรี่ คิดดัลล์ ขณะที่ทำงานเป็นวิศวกรที่บริษัท Intel ในขณะนั้น Intel ได้พัฒนา



p System นั้นแตกต่างจาก PC DOS และ CP/M 86 ค่อนข้างมาก ในขณะที่ CP/M ทำงานกับไมโครโปรเซสเซอร์ ถึง 5 ตัว (คือ 8080,8086,8088 และ Z 80) และ PC DOS ทำงานได้อีกกับ 5 ตัว (คือ Z 8000,68000,8080,8086 และ 8088) p System นั้นสามารถทำงานได้กับระบบ processor และไมโครโปรเซสเซอร์ ถึง 12 แบบด้วยกันคือ

- Intel 8080,8085,8086,8088
- Digital Equipment LS1-11, PDP-11
- Mos technology 6502
- Motorola 6800, 6809, 6800
- Texas instrument 9900
- Zilog Z 80, Z8000
- Western Digital Micro processor

ดังนั้นจึงเห็นได้ชัดว่า โปรแกรมที่เขียนขึ้นภายในระบบนี้จะสามารถทำงานได้กับคอมพิวเตอร์มากชนิดกว่าของ PC Dos และ CP/M 86 หลายเท่าตัวนัก

ตัว P ใน P System ย่อมาจาก pseudo code หรือ Pcode ซึ่งก็คือ Code ซึ่งระบบนี้จะผลิตขึ้นเมื่อโปรแกรมที่เขียนในภาษาต่าง ๆ ภายใต้ระบบนี้ถูก compiled ภาษาที่ใช้ภายใต้ USCD p System ได้ก็คือ PASCAL, FORTRAN, BASIC

เมื่อโปรแกรมในภาษาต่าง ๆ เหล่านี้ถูก Compiled มันจะกลายเป็น P code แทนที่จะเป็นภาษาเครื่อง (machine language) ต่อมา P code จะถูกแปลไปเป็นภาษาเครื่องโดย P-machine emulator ซึ่งเป็นโปรแกรมที่เปลี่ยน P code ไปเป็นคำสั่งที่สามารถ execute ได้โดยตรงโดยไมโครโปรเซสเซอร์

ไมโครโปรเซสเซอร์แต่ละตัวที่

ไมโครโปรเซสเซอร์ 8008 ขึ้นมา ดร. คิลดัลล์กับเพื่อนร่วมงานได้เขียน high level compiler ขึ้นมาตัวหนึ่งมีชื่อว่า PL/M เพื่อให้ทำงานกับชิพใหม่นี้ ในการพัฒนา PL/M ขึ้นมานั้น VERSION เริ่มแรกของ CP/M ก็ได้พัฒนาขึ้นมา แรกที่สุดนั้น CP/M ได้รับการออกแบบมาให้จัดการข้อมูลที่บันทึกอยู่บนเทปกระดาษ โดยตอนนั้น floppy disk ยังมีราคาแพงอยู่มาก และระบบเก็บข้อมูลของไมโครคอมพิวเตอร์ที่แพร่หลายที่สุดก็คือการใช้เทปกระดาษ ต่อมา CP/M ก็ได้รับการดัดแปลงมาให้ใช้กับ floppy disk ได้และยังได้รับเอาโปรแกรมหลายอย่างไว้ด้วยเช่น Text Editor, assembler และ debugger หรือช่วยในการพัฒนาโปรแกรมภาษาเครื่อง

สิ่งหนึ่งที่ทำให้ CP/M ได้รับความนิยมอย่างกว้างขวางก็เพราะมีซอฟท์แวร์ที่เขียนมาให้ทำงานใต้ CP/M อย่างมากมาย และเนื่องจากว่าการเปลี่ยนโปรแกรมจาก CP/M 80 ไป ยัง CP/M 86 นั้นค่อนข้างง่าย จึงเป็นที่คาดกันว่าจะมี Software จำนวนมากสำหรับให้ใช้งานสำหรับไมโครโปรเซสเซอร์ที่ทำงานได้ CP/M

อย่างไรก็ตาม CP/M ก็มีข้อเสียอยู่หลายอย่างเหมือนกัน ประการแรกก็คือหนังสือคู่มือซึ่งผู้ใช้หลายคนพบว่ายากที่จะอ่านเข้าใจ ประการที่สองคือ CP/M ค่อนข้างจะใช้ยากกว่า Operating System อื่น ๆ โดยเฉพาะอย่างยิ่งสำหรับผู้ใช้มือใหม่ และประการสุดท้าย CP/M มีข้อจำกัดหลายประการเมื่อทำงานกับระบบ hard disk ขนาดใหญ่

UCSD p system

UCSD p System พัฒนาขึ้นโดย Institute for Information Science ที่มหาวิทยาลัย California ที่ San Diego ในราวปี 1975 และต่อมาก็ได้ให้ลิขสิทธิ์ในการจำหน่ายแต่ผู้เดียวแก่ Softech Micro System ซึ่งเป็นบริษัทลูกของ Softech Inc.

อยู่ใต้ UCSD-P system จะต้องการ P-machine emulator ที่แตกต่างกัน UCSD P code ใช้คำสั่งชุดเดียวกับไมโครโปรเซสเซอร์ของ Western Digital ดังนั้น P code จะถูก execute โดยไมโครโปรเซสเซอร์ตัวนี้โดยตรงโดยไม่ต้องมี P-machine emulator

Compiler กับ Interpreter

โปรแกรมภาษาที่ Compiled จะประกอบด้วย 2 ส่วนคือ source code และ Compiled code Source code ประกอบด้วยโปรแกรมประโยคคำสั่งต่าง ๆ ที่อยู่ในรูปเดิม ตัวอย่างเช่น

100 INPUT "ENTER TODAY'S DATE:" DATE 1

ซึ่งเป็นภาษา CBASiC sourec code จะถูกแปลโดยโปรแกรมที่เรียกว่า Compiler จนกลายเป็น compiled code ซึ่งจะมีลักษณะใกล้เคียงกับภาษาเครื่องซึ่งใช้โดยไมโครโปรเซสเซอร์ compiled code ก็คือ code ที่ใช้งานจริง ๆ ตอนที่ compile program run อยู่ โปรแกรมที่ RUN Compiled program เรียกว่า ***run time monitor.***

ส่วนภาษาที่ใช้ interpreter นั้นมีแต่ source code source codeจะถูกแปลบรรทัดต่อบรรทัดให้กลายเป็นคำสั่งภาษาเครื่อง ภาษา BA-SIC ที่ใช้กับ IBM PCใช้ Interpreter

ข้อได้เปรียบอย่างหนึ่งของการใช้ Interpreter ก็คือ โปรแกรมสามารถพัฒนาขึ้นมาได้ง่ายเมื่อทำงานกับ Interpretered Language โปรแกรมเมอร์เพียงแต่เขียนโปรแกรม run มันแก้ไขดัดแปลงมันด้วยความสะดวก แต่เมื่อทำงาน ภาษา ที่ compiled source code ต้องได้รับการ compiled ใหม่ทุกครั้งที่มีการ edit ทำให้การเขียนโปรแกรมยุ่งมากขึ้นโดยเฉพาะการ debug โปรแกรม

ส่วนข้อได้เปรียบ Compiled language ก็คือสามารถทำงานได้รวดเร็วกว่าทั้งนี้เพราะว่า compiled code นั้นใกล้กับภาษาเครื่องมากกว่า source code จึงไม่ต้องการแปลอีก การทำงานจึงเร็วกว่าแบบ interpreter มาก

Monochrome Display

จอภาพวิดีโอหรือ CRT เป็นตัวกลางในการสื่อสารระหว่างผู้ใช้และคอมพิวเตอร์ จอภาพสีเดียว(Mono-chrome Display) ของ IBM ใช้จอภาพขาวดำและฟอสเฟอร์สีเขียวเพื่อที่จะทำให้เกิดภาพสีดำและเขียว จอภาพสีดำและเขียวจะช่วยลดความเครียดของนัยน์ตาเวลาที่ต้องใช้งานนาน ๆ

จอภาพของ IBM มีขนาดเท่ากับ TV ขาวดำขนาด 12 นิ้ว

จอภาพจะได้ไฟเลี้ยงมาจาก System Unit สายไฟ AC จากจอภาพจะต่อตรงเข้ากับแหล่งจ่ายไฟทางด้านหลังของ System Unit ทำให้สวิทช์ปิด/เปิดบน System Unit สามารถควบคุมไฟได้ทั้งบน System Unit และบนจอภาพจอภาพจะมีสายเคเบิ้ลสัญญาณวิดีโอ ซึ่งต่อเข้ากับ Monitor/Printer Adapter อยู่บนทางขวาด้านหลังของ System Unit

จอภาพของ IBM สามารถแสดงตัวอักษรได้ 80 ตัว 25 บรรทัด ตัวอักษรจากจอภาพของ IBM จะมีความคมชัดมากและอ่านได้ง่ายทำให้ผู้ใช้ไม่ระคายเคืองนัยน์ตา

ตัวอักษรของจอภาพของ IBM เกิดจาก Matrix ขนาด 7 (แนวนอน) × 9 (แนวตั้ง) อยู่ใน matrix ขนาด 9 × 14 พื้นที่ส่วนอื่นที่เพิ่มเติมมานี้ใช้สำหรับตัวอักษรอังกฤษตัวเล็กที่มีเชิงเช่น y หรือ p เพื่อให้มันดูเป็นปกติ

คุณสมบัติของจอภาพของ IBM PC แสดงไว้ในตารางที่ 8

ตารางที่ 8 IBM Monochrome Display

ขนาด	: สูง 11 นิ้ว (280 มม.) กว้าง 15 นิ้ว (380 มม.) ลึก 14 นิ้ว (350 มม.) หนัก 17 ปอนด์ (7.9 กก.)
จอภาพ	: ขนาดตามแนวทะแยงมุม 11.5 นิ้ว (292 มม.) ใช้ฟอสฟอร์สีเขียว P-39
สายเคเบิ้ล	: AC Power Cable Video Cable (ใช้คอนเนคเตอร์ตัว D มี 9 pin)
คุณสมบัติ	: 1. 80 คอลัมน์ × 25 บรรทัด 2. 7 × 9 Matrix ใน 9 × 14 box 3. Direct Drive Video 4. Screen Refreshmant 50 hz ความคมชัด 720 × 350 จุด

IBM CPS -80 Printer

IBM CP S-80 Printer ที่จริงก็คือเครื่องพิมพ์ Epson MX-80 นั่นเอง เครื่องพิมพ์นี้สามารถพิมพ์ตัวอักษรได้ด้วยความเร็ว 80 ตัว ต่อวินาที และด้วยแบบตัวอักษรหลายแบบ

เครื่องพิมพ์ IBM นี้จะพิมพ์แบบฟอร์มได้ด้วยความกว้างต่ำสุด 4 นิ้ว และกว้างสุด 10 นิ้ว ความหนาของกระดาษที่หนาสุดที่ยอมรับได้คือ 3 plies หรือ 0.012 นิ้ว

กระดาษจะถูกป้อนเข้าเครื่องพิมพ์จากทางด้านหลัง หมึก ribbon จะอยู่ใน catridge ที่สามารถถอดเปลี่ยนได้ ribbon แต่ละชุดสามารถใช้พิมพ์ได้ประมาณ 3 ล้านตัวอักษรหรือเท่ากับการพิมพ์ต่อเนื่องกันประมาณ 10 ชั่วโมง

เครื่องพิมพ์ IBM สามารถพิมพ์ได้ 2 ทาง (bidirectional) ด้วยหัวพิมพ์ 9 จุดหัว พิมพ์นี้มีอายุประมาณ 30 ล้านครั้ง และสามารถถอดเปลี่ยนได้โดยผู้ใช้เองถ้าชำรุด

ตัวพิมพ์จะเกิดจาก matrix 9 × 9 ตัวหนังสือสามารถเปลี่ยนขนาดได้ตั้งแต่ 5 ถึง 16.5 character per inch (CP I) บนความกว้าง 8 นิ้ว โดยการปรับ pitch size (ปรับได้ 5, 8.25, 10 และ 16.5) จะทำให้พิมพ์ตัวอักษรได้มากสุด 40, 66,80 และ 132 ตัวตามลำดับ

เครื่องพิมพ์ IBM สามารถพิมพ์ตัวอักษรได้ 2 แบบ คือแบบธรรมดา, พิมพ์ซ้ำ (double) และแบบเน้น (emphasized) *การพิมพ์แบบธรรมดาคือการที่หัวพิมพ์กระแทกลงบนกระดาษพิมพ์เพียงครั้งเดียวต่อหนึ่งตัวอักษร การพิมพ์ซ้ำคือการพิมพ์ตัวอักษรนั้น 2 ครั้งลงไปยังตำแหน่งเดียวกัน ส่วนการพิมพ์เน้นเป็นการพิมพ์ซ้ำ 2 ครั้งเหมือนกันแต่เมื่อพิมพ์ครั้งแรกไปแล้วกระดาษจะขยับขึ้นเล็กน้อยก่อนนี้จะพิมพ์ครั้งที่ 2 ลงไป การพิมพ์ที่จะทำตัวอักษรแลดูหนาขึ้น*

เครื่องพิมพ์อื่น ๆ นอกเหนือจากเครื่องพิมพ์ IBM CPS-80 อาจนำมาใช้กับ IBM PC ได้ Parallel Printer อาจนำมาต่อได้โดยใช้ Monochrome Display/printer Adapter หรือ Printer Adapter Card Serial Printer สามารถนำมาต่อโดยใช้ Asynchronous Communications Adapter เข้าช่วยด้วย

ตารางที่ 9 คุณสมบัติของ IBM CPS 80

ผู้ผลิต	: Epson America Inc.
หมายเลขโมเดล	: MX 80
ปีออกสู่ตลาด	: 1979 โดย Epson
ขนาด	: สูง 4.2 นิ้ว (107 มม.) กว้าง 14.2 นิ้ว (374 มม.) ลึก 12.0 นิ้ว (305 มม.) หนัก 12 ปอนด์ (5.5 กก.)
ความเร็ว	:80 CPS (charactre per second)
Interface	: Parallel (Centronic Standard)
ขนาดกระดาษ	: กระดาษแบบมีรูหนามเตย กว้างต่ำสุด 4 นิ้ว (102 มม.) กว้างมากสุด 10 นิ้ว (254 มม.) หนาสุด 3 plies (0.012 นิ้ว)
คุณสมบัติ	: 1. หัวเป็นแบบ 9 pin 2. พิมพ์ได้ 2 ทาง (ไปกลับ) มี logic seeking 3. ตัวอักษร 9 × 9 ตัวอักษร ASC II 96 ตัว สัญลักษณ์ 9 ตัว สัญลักษณ์กราฟิค 64 ตัว 4. ใช้ ribbon cartridge สีดำ พิมพ์ได้ประมาณ 3 ล้านตัวอักษร 5. มีเซนเตอร์ตรวจกระดาษหมด 6. สามารถทดสอบตัวเองได้ (Self test)
คุณสมบัติที่คุมด้วยซอฟท์แวร์	: 1. ขนาดพิมพ์

ชื่อ	Pitch	พิมพ์ได้มากสุด (ตัว)
Normal	10	80

Enlarged	5	40
Condensed	16.5	132
Contensed	8.25	66

2. แบบของการพิมพ์

Normal	พิมพ์ครั้งเดียว
Double	พิมพ์ซ้ำ 2 ครั้ง
Emphasized	พิมพ์ครั้งที่ 1 เลื่อนกระดาษขึ้น 1/216 นิ้วพิมพ์ครั้งที่ 2

3. ระยะห่างบรรทัด ปรับได้เป็น 6, 8, 10 บรรทัดต่อนิ้ว
4. ตั้ง tab ได้ทั้งแนวตั้งและแนวนอน

Serial และ Parallel communication

ข้อมูลอาจจะส่งจากคอมพิวเตอร์ไปยังเครื่องรับ (เช่นเครื่องพิมพ์หรือจอภาพ) ด้วย 2 วิธีด้วยกันคือแบบอนุกรม (Serial) และแบบขนาน (Parallel) ในการส่งแบบขนานนั้นเราสามารถที่จะส่งข้อมูลได้ทีละ 8 บิท ในเวลาเดียวกัน แต่ในการส่งแบบอนุกรมเราส่งได้เพียงทีละบิทในเวลาเดียวกัน

Monochrome Display/ Printer Adapter Card

Monochrome Display/ Printer Adapter Card สามารถเชื่อม PC เข้ากันทั้งจอภาพสีเดียวและเครื่องพิมพ์ได้ในเวลาเดียวกัน

ส่วน Printer Adapter Card นั้นทำหน้าที่เพียงส่วนเดียวของ Card แรกคือเชื่อม PC เข้ากับเครื่องพิมพ์เท่านั้น

ข้อแตกต่างเพียงอย่างเดียวระหว่าง Display/Printer Adapter กับ Printer Adapter Card อยู่ที่ตำแหน่งของ Memory location กล่าวคือ Memory location เริ่มต้นของ Card ทั้งสองจะอยู่ต่างกัน 68 location นอกนั้นการทำงาน (ในส่วน Printer) ก็เหมือนกันทุกอย่าง

ถึงแม้การ interface กับเครื่องพิมพ์ IBM จะใช้มาตรฐานของ Centronic ที่จุดต่อระหว่างเครื่องพิมพ์ไปยัง interface card จะให้ connector DB-25 ในการต่อเครื่องพิมพ์เข้ากับ PC นั้นต้องใช้สาย Printer Cable ของ IBM ซึ่งสาย Cable นี้มีปลายหนึ่งเป็น DB-26 connector อีกปลายหนึ่งเป็น Contronic connector DB-26 จะใช้ต่อกับ system Unit ในขณะที่ Centronic connector จะต่อกับเครื่องพิมพ์ สาย IBM Printer Cable นี้สามารถใช้ต่อกับเครื่องพิมพ์ชนิด Parallel ชนิดใดก็ได้นอกเหนือจากของ IBM

การที่ใช้ DB-25 connector กับ PC ก็เพื่อที่จะป้องกันปัญหาที่เรียกกันว่า Crosstalk Crosstalk เป็นปัญหาที่พบได้ทั่วไปเมื่อใช้สายเคเบิ้ลยาว ๆ กับระบบสื่อสารแบบขนาน Crosstalk เกิดขึ้นเมื่อสัญญาณไฟฟ้าของสายสัญญาณต่าง ๆ เกิดการรบกวนกัน เพื่อที่จะลด Crosstalk จึงต้องมีการเพิ่มสายขึ้นมาเพื่อ ground สัญญาณรบกวนเหล่านี้ลดน้อยลงไป

Display/Printer card มี ROM ขนาด 8K อยู่ด้วย (สำหรับบรรจุ code สำหรับตัวอักษรประมาณ 256 ตัว) และ RAM ขนาด 4K สำหรับจุข้อมูลที่เป็นตัวอักษรได้ประมาณ 1 หน้า RAM จะถูก refreshed 50 ครั้งทุก ๆ วินาที

ในการแสดงตัวอักษร 1 ตัวบนจอภาพนั้น Display/Printer Adapter ต้องการข้อมูลจำนวน 2 byte byte แรกนั้นสำหรับแสดง ASC II code สำหรับตัวอักษรนั้น

ส่วน byte ที่ 2 เรียกว่า attribute code ในจอภาพสีเดียว attribute code จะควบคุมการทำงาน 4 อย่างคือ

1. ความมืดของ background
2. ความมืดของ foregroumd
3. ความเข้มหรือความสว่างของตัวหนังสือ
4. ตัวหนังสือกะพริบ หรือไม่กะพริบ

ซึ่งเมื่อนำมาผสมกันจะได้ลักษณะบนจอภาพสีเดียวถึง 6 อย่างคือ

1. ตัวหนังสือสีขาวบน backgroud สีดำ (video mode)
2. ตัวหนังสือสีดำบน back ground สีขาว (reverse video)
3. ตัวหนังสือสีขาวกะพริบบน foreground สีมืด
4. ตัวหนังสือสีขาวบน back-ground สีขาว (invisible character)
5. ตัวหนังสือสีดำบน bacdgroud สีดำ
6. ตัวหนังสือสีดำกะพริบบน background สีขาว

ดังได้กล่าวแล้วข้างต้นทั้ง Mono

IBM Monochrome Display/Printer Adapter	
Size:	Height: 4 in. 102 mm Width: 14 in. 357 mm Depth: .25 in. 6.3 mm
Location:	System Unit Expansion Slot (generally slot 2)
Characteristics:	1. Display a. 80 x 25 or 40 x 25 text modes b. Normal, reverse, flashing, high-intensity modes c. Motorola 6845 CRT controller d. 4K RAM display buffer e. 8K ROM for 256 character codes 2. Printer a. 8 bit parallel port (Centronics compatible) b. Acknowledge, Busy, Paper Out, and Select signals c. Allows use of any Centronics compatible printer device d. 12 TTL buffered output points e. 5 input ports readable under program control f. 25 pin D-connector (female) used

chrome และ Color Adapter สามารถแสดงได้ 25 บรรทัด โดยทั่วไปแล้ว โปรแกรมจะใช้อย่างมากก็ 24 บรรทัดเท่านั้น บรรทัดที่ 23 จะใช้เป็น Status line สำหรับใส่ prompt หรือข่าวสารต่าง ๆ ให้กับผู้ใช้งานเครื่องได้มาก

Monochrome Display/Printer Adapter ใช้ CRT controller เบอร์ 6845 ของ Motorola ในการ interface กับจอ video (เช่นเดียวกับ (Color/Graphics Monitor Adapter) 6845 สามารถควบคุมได้ด้วย oftware หมายความว่าโปรแกรมสามารถควบคุม output ของจอได้โดยตรง ตัวอย่างเช่นโปรแกรมหนึ่งสามารถจัดจอภาพให้มีขนาดเล็กกว่า 25 × 80 ได้

Color Graphic Monitor Adapter

ข้อแตกต่างสำคัญระหว่าง Color / Graphic Monitor Adapter และ Display/printer Adapter คือ Color/Graphic board สามารถแสดงสีได้ ในขณะที่ Display/printer board ทำไม่ได้

Color/Graphic Monitor Adapter สามารถใช้กับ Color monitor หรือ TV สีโดยใช้ RF Modulator เข้ามาช่วย

Color/Graphic Adapter ทำงานได้ 2 วิธีคือ alphanumeric (A/N) หรือ text mode และ all point addressable (APA) graphic Mode แต่ละ Mode ที่กล่าวมายังมี Mode ย่อย ๆ แยกออกไปอีก Mode

A/N Mode

A/N Mode จะให้ Video output เป็น 40 ตัวอักษร × 25 บรรทัด เมื่อใช้กับเครื่องรับโทรทัศน์หรือ monitor ความละเอียดต่ำหรือความละเอียดสูง Video output ขนาด 25 บรรทัด × 80 ตัว อักษรทำได้โดยใช้ monitor ความละเอียดสูง

ใน A/N Mode ของจอสีเดียว ตัวหนังสือสามารถแสดงเป็นแบบ reverse, high intensity และกะพริบได้ ใน Color Mode foreground หรือตัวหนังสือสามารถแสงได้ 16 สีบน background สีหนึ่งสีใดจาก 8 สี ตัวหนังสือสามารถกะพริบได้

สีต่าง ๆ ที่สามารถใช้กับ Color/Graphic Adapter จะแสดงไว้ในตารางที่ 8 ตัวเลขที่ใช้บ่งบอกสีใน BASIC แสดงไว้ในคอลัมน์ที่ 1

สีทั้ง 16 ได้มาจากการผสมของแม่สีคือ red, green และ blue และสัญญาณบอกความเข้ม (intensity signal) การผสมสีทางอิเลคทรอนิคส์แตกต่างจากการผสมสีวาดภาพ โดยแม่สีที่ 3 คือสีเขียวไม่ใช่สีเหลืองตามแม่สีสำหรับวาดภาพ สี 8 ถึง 15 ในตารางที่ 8 อาจจะไม่สามารถแสดงออกมาได้ใน monitor หรือ TV ซึ่งไม่สามารถจับสัญญาณ intensity control ได้

Color No.	Color Name	Red	Green	Blue	Intensity Signal
0	Black	N	N	N	N
1	Blue	N	N	Y	N
2	Green	N	Y	N	N
3	Cyan	N	Y	Y	N
4	Red	Y	N	N	N
5	Magenta	Y	N	Y	N
6	Brown	Y	Y	N	N
7	Light Grey	Y	Y	Y	N
8	Dark Grey	N	N	N	Y
9	Light Blue	N	N	Y	Y
10	Light Green	N	Y	N	Y
11	Light Cyan	N	Y	Y	Y
12	Light Red	Y	N	N	Y
13	Light Magenta	Y	N	Y	Y
14	Yellow	Y	Y	Y	Y
15	White	Y	Y	Y	Y

Y = Yes N = No
Colors 8-15 may not be available on some monitors & TV s.

APA MODE

APA (All Point Addressable) dode ทำให้โปรแกรมสามารถควบคุมทุกจุดบนจอภาพได้ APA Mode ยังแบ่งออกเป็น 3 Mode คือ high resolution mode, medium resolution mode ,low resolution mode

ใน high resolution mode จอจะถูกแบ่งเป็น 640 จุดตามแนวนอน และ 200 จุดตามแนวตั้ง Medium resolution ประกอบด้วย 320 จุดในแนวนอนและ 200 จุดตามแนวตั้ง และ low resolution ประกอบด้วย 160 จุดในแนวนอนและ 200 จุดในแนวตั้ง

ตารางที่ 10 IBM Asynchronous Communications Adapter

Size:	Height: 4 in. 102 mm Width: 5 in. 127 mm Depth .25 in.
Components:	1. INS 8250 Asynchronous Communications Element (ACE) 2. ACE Support Circuiting 3. DB-25M connector on card extending from rear of System Unit. 4. Jumper block for RS-232C or 25 ma. current loop (TTY compatible)
Characteristics:	1. Fully software controllable 2. 50-9600 baud

จอภาพสำหรับ Color/Graphic Monitor Adapter

จอภาพที่ใช้กับ Color/Graphic Monitor Adapter ได้มี 4 แบบ คือ TV ธรรมดา monitor ขาวดำ, monitor สีและ RGB color monitor

RF Modu lator จำเป็นต้องใช้เมื่อต่อกับ TV สีเพื่อเปลี่ยนสัญญาณวิดีโอ ให้สัญญาณ RF ซึ่งทำให้เข้า TV ทางสายอากาศได้

เมื่อใช้กับจอ TV จะใช้ได้เพียง mode 40 × 25 เท่านั้น ไม่สามารถแสดงตัวหนังสือแบบ 80 × 25 ออกมาได้

Asynchronous Communication Adapter

Asynchronous Communication Adapter ทำให้ IBM PC สามารถติดต่อสื่อสารกับเครื่องมืออื่น ๆ ได้ เช่น คอมพิวเตอร์เครื่องอื่น เครื่องพิมพ์และ Modem

Asynchronous Communication เป็นวิธีการส่งข้อมูลแบบหนึ่ง ซึ่งข่าวสารจะถูกส่งไปยังเครื่องรับทีละหนึ่งบิทพร้อมกับสัญญาณเพิ่มเติมเพื่อบอกจุดเริ่มและจุดสิ้นสุดของแต่ละตัวอักษร หรือดูอีกนัยหนึ่ง Asynchronous Communication ประกอบด้วยการส่งข้อมูลแบบอนุกรมไปยังเครื่องรับโดยมีช่วงจังหวะเวลาที่ไม่แน่นอน Asynchronous Communication Adapter สามารถใส่ลงใน Slot ไหนใน 5 Slot ก็ได้ ตัว Adapter จะมี DB 25 (25 pin) connector ซึ่งต่อมาจากด้านหลังของ System Unit คุณสมบัติของ Asynchionous adapter ได้แสดงไว้ในตารางที่ 10

(อ่านต่อฉบับหน้า)

ประวัติและแนวการจัดมาตรฐานรหัสอักษรไทย

โดย พิสุทธิ์ สภาพรภูริศักดิ์

แรงดลใจอย่างหนึ่งที่ทำให้เขียนเรื่องนี้ก็คือ *การแสดงผลิตภัณฑ์ทางไมโครคอมพิวเตอร์ในงานอีเล็คตรอน-83* ที่สวนอัมพรเมื่อเร็ว ๆ นี้ สิ่งที่น่าสนใจมากก็คือการที่เครื่องคอมพิวเตอร์หรืออุปกรณ์ต่าง ๆ สามารถสำแดงผลงานออกมาเป็นภาษาไทยกันมากขึ้นแต่ละระบบแต่ละอุปกรณ์ก็ให้รายงานออกมาแตกต่างกัน เช่น บางระบบมีแต่เฉพาะการแสดงผลและเก็บข้อมูลได้เท่านั้น ยังพิมพ์ออกมาไม่ได้ บางระบบก็มีพร้อมชุดคือแสดงผลบนจอภาพได้เก็บรักษาได้และพิมพ์ผลออกมาได้

ปัญหาของผู้ซื้อ

ผู้ซื้อมักจะไม่เข้าใจปัญหาเพราะคิดว่าถ้ามีอักษรภาษาไทยในเครื่องแล้วทำอะไรได้ทุกอย่าง, แต่ที่จริงไม่ได้เป็นเช่นนั้น ส่วนมากแล้วระบบอักษรไทยสามารถใช้ได้กับภาษาทั่วไปได้ เช่น BASIC, COBOL, FORTRAN หรือภาษารวมรหัส (assembly) ส่วนโปรแกรมชุดงาน เช่น MULTIPLAN, LOTUS 123 dBASE II ฯลฯ นั้นอยู่ที่ว่าได้ถูกแก้ไขโปรแกรมให้ใช้กับอักษรไทยได้แล้วหรือยัง ดังนั้นจึงเป็นหน้าที่ของผู้ซื้อที่จะต้องสืบถามถึงความเป็นมาของสิ่งที่ได้ เช่น เครื่องและภาษา รวมทั้งการที่จะใช้ hardware และ software ให้ได้เป็นผลตามข้อกำหนดและความต้องการของสำนักงาน

ความต้องการครอบจักรวาล

สิ่งที่บริษัทผู้ผลิตหรือจำหน่ายประสงค์จะให้เครื่องของตนสามารถทำงานได้กับข้อมูลภาษาไทย กล่าวคือต้องการให้ทำงานได้ทุก ๆ อย่าง ทุกรูปแบบของธุรกิจ เลยมาถึงการทำ word processing และการทำฐานข้อมูลของระเบียนประวัติต่าง ๆ ทั้งที่เกี่ยวและไม่เกี่ยวกับธุรกิจ แต่ที่จริงงานทางด้านภาษาไทยที่ยังต้องการความช่วยเหลือจากเครื่องคอมพิวเตอร์ยังมีอีกมาก เช่น การค้นคว้าเกี่ยวกับการออกเสียง, การตรวจสอบและทำพจนานุกรม, การสังคายนาบทธรรม, ด้านการประพันธ์ เป็นต้น

หน้าที่ของใคร ?

ถ้าใครอยู่ในบริษัทที่จำหน่ายเครื่องคอมพิวเตอร์ คงจะนึกถึงภาพเหตุการณ์สมัยแรก ๆ ออกว่าเกิดความโกลาหลเช่นใด เมื่อทางบริษัทต้องการให้คอมพิวเตอร์ฝรั่งพิมพ์ภาษาไทยได้, และต้องการให้ได้ผลตรงตามธรรมชาติและความงามตามลักษณะอักษรไทย คงจะไม่มีบริษัทไหนสั่งให้ลูกน้องทำให้เครื่องฝรั่งพูดไทยด้วยนะครับ (แต่เดี๋ยวนี้เครื่องทั้งฝรั่ง, จีน, ไทย พูดไทยได้จ้อแทบฉีกผ้าอ้อมไม่ทัน) เท่าที่จำได้ผู้เขียนเคยพบการออกแบบอักษรไทยสำหรับคอมพิวเตอร์ครั้งแรกที่ *บริษัทคอนโทรลดาต้า*ตึกธนาคารเชสแมนฮัตตัน ในโรงแรมดุสิตธานี เมื่อปีพ.ศ. 2519 (ถ้าใครมีประสบการณ์มากกว่านี้กรุณาส่งบันทึกช่วยความจำมาด้วยนะครับ) เขาออกแบบกันทั้งจุดภาพและแบบตัวพิมพ์ธรรมดา ด้านออกแบบลายอักษรเป็นหน้าที่ของฝ่าย software ส่วนการเปลี่ยนแปลงวงจรให้ได้ผลการพิมพ์ตามประสงค์ ฝ่าย hardware เป็นผู้ทำ อักษรลายจุดนั้นเขาทำได้ง่ายเพราะเพียงแต่ทำโปรแกรมส่งรหัสลาย

จุดเข้าเครื่อง ๆ ก็พิมพ์อักษรไทยออกมาได้ ส่วนตัวพิมพ์แบบธรรมดานั้นเขาจะส่งภาพลายอักษรไทยตัวโต ๆ ไปย่อลงฟิล์มเพื่อทำลายบนแถบเหล็กสปริงที่ใช้เป็นเข็มขัดพิมพ์อักษร ทุกบริษัทคงมีกระบวนการของตนเองดังที่ว่าและหน้าที่ก็คงเป็นของวิศวกรคอมพิวเตอร์และนักโปรแกรมระบบที่จะพิจารณาเทคนิค ในการใช้งานปัจจุบันคอมพิวเตอร์ที่มีชื่อเกือบทุกยี่ห้อมีอักษรไทยติดตั้งในเครื่องแล้ว, แต่จะเป็นที่พอใจ ตัวอักษรสวยหรือไม่ หรือได้มาตรฐานเท่าใดนั้นอยู่ที่วิจารณญาณของผู้ใช้เอง

เวรอยู่กับคนทำ กรรมอยู่กับคนใช้

กลุ่มคนทำทั้งฝ่าย hardware และ software เมื่อได้รับคำสั่งให้ทำก็ทำจนเต็มความสามารถอยู่หลายเดือนกว่าที่เครื่องจะแสดงผลเป็นภาษาไทยได้ เมื่อแก้ไขเสร็จเรียบร้อยฝ่ายผู้ใช้ก็หวังที่จะได้เห็นอักษรไทยตามลักษณะที่เป็นอยู่ แต่ไม่เป็นเช่นนั้นเพราะระบบสากลเก็บข้อมูลเรียงกันเป็นแถวทั้งในหน่วยจำและในจานบันทึก ดังนั้นเมื่อแสดงผลออกมาทางจอภาพหรือเครื่องพิมพ์อักษรไทยจะเรียงกันเป็นแนวเดียวอย่างที่มีศัพท์สมัยแรก ๆ ว่า "ภาษาขอม" จึงเป็นหน้าที่ของฝ่าย software ที่จะต้องเขียนโปรแกรมที่จะให้อักษรไทยอยู่ในตำแหน่งที่ถูกต้อง

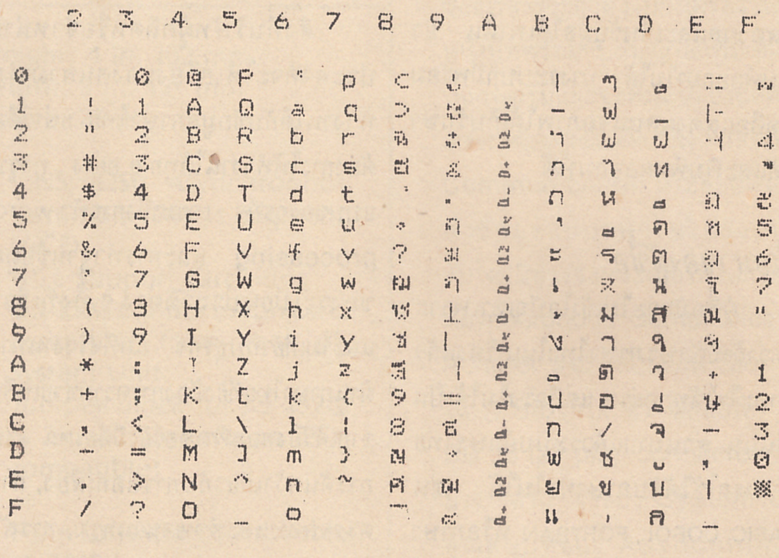
ส่วนผลการดำเนินงานจะเปลืองเวลา, ทุน, และบุคคลากรมากน้อยเท่าใดนั้นขึ้นอยู่กับการตัดสินใจเลือกรหัสที่ใช้รวมทั้งผลกำไร และความปลอดภัยของตนก็เป็นส่วนสำคัญ

ชาตะและมรณะ

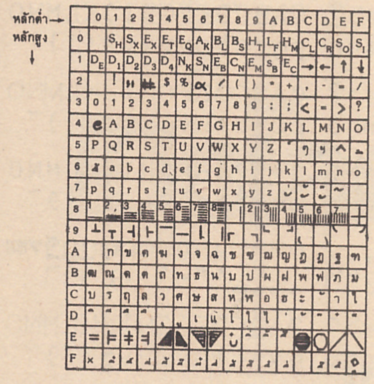
ทุกสิ่งย่อมมีเกิดและตายไปตามธรรมชาติ รหัสอักษรไทยก็เช่นเดียวกันต่างก็เกิดและตายเป็นรุ่น ๆ (คล้าย ๆ กับพระเครื่องอย่างนั้นแหละครับ) ดังนั้นก่อนที่จะยกตัวอย่างเพื่อสาธยายอิทธิปาฏิหาริย์กัน ก็ขอตั้งศัพท์แสงเสียหน่อยตามธรรมเนียม ขอใช้คำว่า "ผลิตอัตราการ (generation)" หรือเรียกย่อ ๆ ว่า "รุ่น" กับรหัสชุดหนึ่ง ๆ ที่มีการเปลี่ยนแปลงไปคือ "เกิด" และไม่ถูกใช้อีก ("ตาย") แต่ก็ยังมีปัญหาความต่อไปอีกว่ารหัสชุดใหม่ต้องมีการเปลี่ยนแปลงจำนวนกี่ลักษณะ (character) หรือ "ตัว" จึงจะได้ชื่อว่าเป็นอักษรชุดใหม่ ขอประมาณเอาว่า 5 ตัวก็แล้วกัน ถ้าน้อยกว่านี้ถือว่าเป็นการปรับปรุง จากนี่แล้วก็ขอเบิกตัวอักษรแต่ละรุ่นมาดูถึงความวิจิตรพิศดารกันต่อไป

ตัวอย่างอักษรไทยและรหัส

ในรูปที่ 1 เป็นตัวอย่างรหัสและลวดลายอักษรชุดแรกของบริษัท COMEDP ซึ่งใช้เครื่อง FUJITSU จากตัวอย่างนี้จะสังเกตเห็นว่ารหัสอักษรไทยเริ่มตั้งแต่ 82H ถึง FEH รหัสที่ใช้ 82H คือ ฉ อักษรต่อไปคือ ฮ ฯลฯ เทคนิคสำคัญของอักษรชุดนี้ใช้คือการลดระดับสระกับวรรณยุกต์ลงมารวมกันเป็นบรรทัดเดียว ดังที่จะเห็นได้ในคอลัมน์ที่ 8 และ 9 เริ่มต้นด้วย ั่ และ ิ้ ซึ่งวิธีนี้ทำให้สามารถประหยัดเนื้อที่ของหน้ากระดาษไปได้มาก นอกจากนั้นผู้ออกแบบยังได้สอดแทรก graphic ซึ่งใช้ในการพิมพ์ตารางต่าง ๆ ได้ชิดกันไว้ในพื้นที่นี้ด้วย ทำให้การใช้รหัสซึ่งมีขีดจำกัดเพียง 256 ตัวมีประโยชน์มากขึ้น ลักษณะที่สำคัญของอักษรชุดนี้คือความงามของอักษรผู้ออกแบบสามารถจัดสัดส่วนของอักษรเสียใหม่ให้หางและฐานลายประกอบบางตัวรวมอยู่เข้าไปในตัวอักษรด้วย ทำให้ความรู้สึกผิดจากปกติธรรมชาติลดน้อยลงไป นอกจากนั้นอักษรส่วนมากไม่ทำให้รู้สึกเต้นหรือหลอกเพราะรอยจุดที่ประกอบเป็นตัวอักษรอยู่ อักษรชุดนี้ขอตั้งชื่อว่า COMEDP รุ่น 1 ก็แล้วกัน ส่วนรุ่นต่าง ๆ กันต่อมาคงจะเปลี่ยนแปลงไปบ้างตามความเหมาะสม



รูปที่ 1 อักษรไทยรุ่นแรกของ *Fujitsu* ของบริษัท *COMEDP*



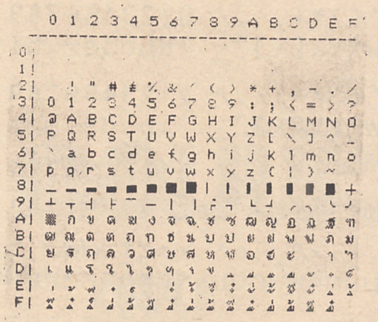
รูปที่ 2 รหัสอักษรไทยแบบ *KTU* รุ่น 2

รหัสและรูปอักษรไทยชุดต่อไปพัฒนาขึ้นในคณะวิศวกรรมศาสตร์มหาวิทยาลัยเกษตรศาสตร์ ซึ่งมีชื่อว่า KTU ตามรูปซึ่งแสดงในรูปที่ 2 แต่รุ่นที่เก่ากว่านี้ยังมีอีกรุ่นหนึ่ง จึงขอตั้งหรือเรียกง่าย ๆ ว่า KTU รุ่น 1 ส่วนแบบซึ่งแสดงในรูปที่ 2 นี้คือ KTU รุ่น 2 รุ่น 1 มีการเรียงของอักษรเหมือนในรูปที่ 2 ทุกประการ ยกเว้นสระที่ตามหลัง ฮ คือ ะ, เ-, แ, โ, ใ, ไ, △, △,ั,ั, △△, ่, ้, ๊, ๋, △, ็, ฺ (△ คือที่ว่าง) จากนั้นจึงมาต่อที่F6H คือิ,ี,ึ,ื และมีนอกช่วงอีก 3 ตัวคือ ุ (5CH), า (7CH) และำ (FEH) การหารหัสของ KTU ชุดหลังนี้ผู้เขียนเองไม่มีตารางนอกจากแกะเอาจาก EPROM. และมีความรู้สึกว่าคล้ายหาธรรมศัพท์ในพระไตรปิฎก รหัส**ชุดนี้**เป็นชุดแรกที่ ใช้ใน dBASE II และ TBASE ในขณะนี้

KTU มีหลักในการเรียงรหัส (ชุดแรก) ดังนี้คือ

1. เรียกลำดับตั้งแต่ ก. ถึง ฮ. เป็น set ไม่ปะปนแบบดาวกระจายหรือจับฉลากเอา

2. ใส่ ฤ หลัง ร โดยไม่มี ฦ หลัง ล ทั้ง 2 ข้อข้างบนเป็นหลักการวางพยัญชนะของ KTU ทุกแบบ



รูปที่ 3 รหัสอักษรไทยแบบ *KTU* รุ่น 3

การออกแบบรหัสของสระและวรรณยุกต์นั้น เขาออกแบบเป็นชุด ๆ เช่นเดียวกัน สำหรับ KTUรุ่น 1 สระชุดแรกเริ่มตั้งแต่ CCH คือ ๆ, ฯ, ะ, เ, แ, โ, ใ, ไ ส่วนชุดที่ 2 เริ่มที่ F6H คือิ,ี, ึ, ื, ตามลำดับ ส่วนชุดที่ 3 เป็นชุดวรรณยุกต์ คือ ่, ้, ๊, ๋ โดยเริ่มที่ DAH และนอกนั้นเป็นสระผสมวรรณยุกต์

ส่วนการออกแบบสระและวรรณยุกต์ของ KTU ตามรูปที่ 2 ซึ่งขอตั้งชื่อว่ารุ่น 2 (ผิดถูกอย่างไรก็ขออภัยล่วงหน้าไว้ก่อนนะครับ) รุ่นนี้จะเรียงสระตามลำดับคือ ะ แทรกด้วย ั ระหว่าง า ตัวต่อมาคือ ไ,ิ,ี,ึ,ื,ุ,ู จากนั้นจึงติดตามด้วยลูกน้องไม้หน้า (ไ) คือ เ, แ, โ, ใ, ไ แล้วจึงถอยร่นไม้คุมเสียงไป 1 หลักคือ ่, ้, ๊, ๋, ไปพบทัพหลังปิดขบวนคือไม้ไต่คู้ จากนนจึงเป็นสระผสมวรรณยุกต์ สระผสมวรรณยุกต์ทีอยู่นอกสเป็คตรัมอักษรไทยเริมจาก 7BH คือ ั่, ั้, ั๊ (ไม่มี ั๋)

ไหน ๆ ก็กล่าวถึง KTU รุ่น 2 ไปแล้วก็จะขอกล่าวถึง KTU รุ่น 3 ต่อไปเลยนะครับ KTU รุ่น 3 นี้คล้ายรุ่น 2 ทุกประการตรงพยัญชนะส่วนสระมีการโยกย้ายกำลังพลกันเล็กน้อย คือ ะ, △, า ยังคงเดิมไว้แล้วนำ ั า, เ, แ, โ, ใ, ไ, ๆ, ฯ, ุ, ู มาต่อไว้ก่อนจากนั้นจึงตามด้วยชุดพินทุ์อิ คือิ,ี,ึ,ื ส่วนนางอายสามพระหน่อคือ ็, ั, ํ, ถูกนำมาต่อท้ายอีกทีเป็นการนำขบวนวรรณยุกต์คือ ่, ้, ๊, ๋ และการันต์ (ชุดนี้ท่าทางขลังมากตามขบวนทัพอักขระที่จัด) หลังจากนั้นเป็นส่วนของสระผสมวรรณยุกต์ทั้งสิ้น

จะเห็นว่าการจัดแบบนี้ทำให้การสรรค์(sort) หรือการเสาะ(search) กระทำได้ง่ายและเป็นระเบียบ ดังนั้นลำดับของการสรรค์ (sort) จะกระทำตามลำดับพยัญชนะ พยัญชนะ+สระโดด พยัญชนะ + วรรณยุกต์ และพยัญชนะ + สระ + วรรณยุกต์ **ตามลำดับ**

การพิมพ์ผลของรหัสอักษรไทยแบบ TKU

การพิมพ์ TKU กระทำได้สะดวกทั้ง 2 แบบ เพราะได้ผ่านการออกแบบและวางรหัสไว้ก่อนเป็นอย่างดี ขอยกตัวอย่าง TKU รุ่น 3 ตรรกกิจ (logic) ของคำสั่งสามารถอธิบายง่าย ๆ ได้ดังนี้คือ.....

ถ้ารหัสของอักษรใดที่มีค่าน้อยกว่า 215 จะถูกพิมพ์ที่บรรทัดกลาง

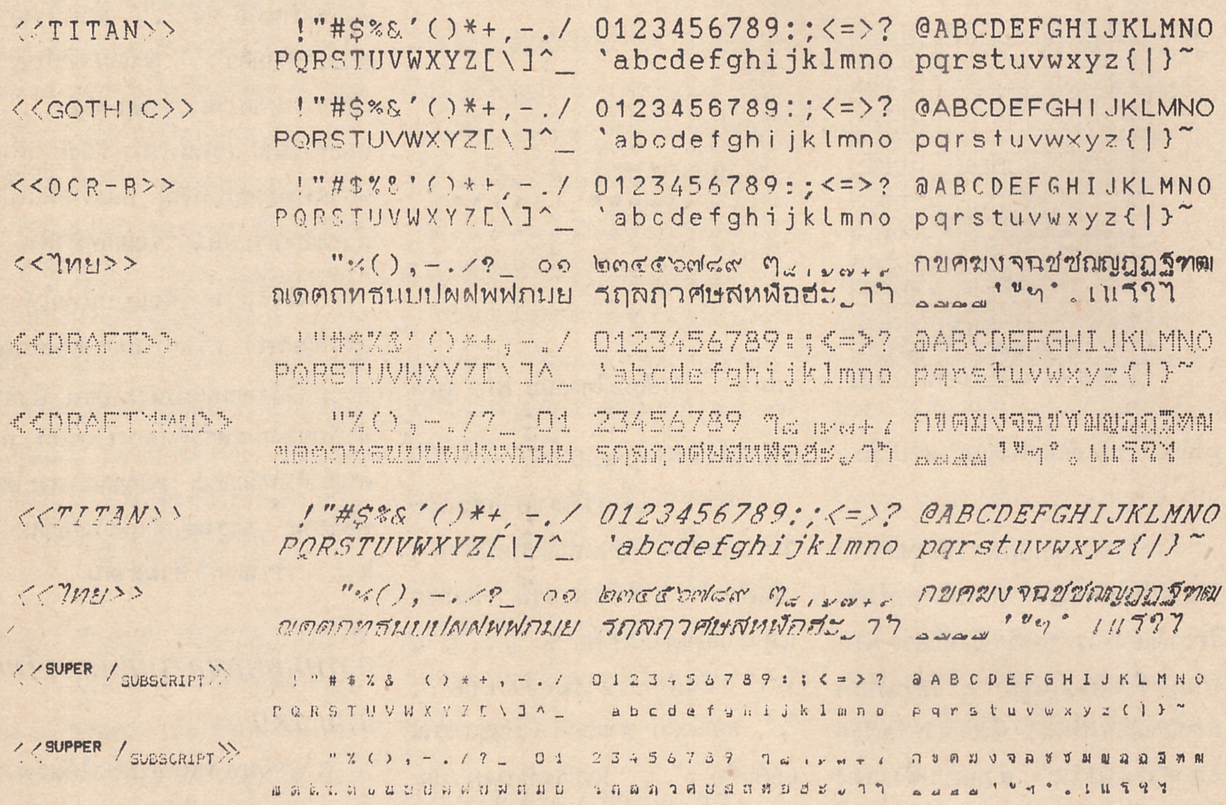
ถ้ารหัสของอักษรใดมีค่าเท่ากับ 215 หรือ 216 จะถูกพิมพ์ที่บรรทัดล่าง

ถ้ารหัสของอักษรใดมีค่ามากกว่า 229 จะถูกพิมพ์ที่บรรทัดบน

จะเห็นว่าตรรกกิจแบบ TKU ในการส่งอักษรไปพิมพ์เปลืองหน่วยจำน้อยมาก, ผิดกับรหัสแบบดาวกระจายที่จะใช้หน่วยประจำประมาณ 20% ของหน่วยจำงานทั้งหมด (กรณีหลังนี้ไม่ผู้ขายก็ผู้ซื้อมหา'ลัยกันไปข้างหนึ่ง)

น้องใหม่ของวงการ

รหัสชุดใหม่เอี่ยมซึ่งแสดงในรูป



รูปที่ 4 การเรียงของอักษรและรหัสแบบ *SMC*

ที่ 4 นี้เป็นของบริษัทสหมิตรเครื่องกล ขอตั้งชื่อเลยนะครับว่าเป็นแบบ SMC รุ่นแรก ได้ข่าวจากผู้ชมในงานว่าทางบริษัทออกแบบแล้วให้ช่างที่อเมริกาผู้ผลิตอุปกรณ์เป็นผู้โปรแกรมให้จึงทำให้ค่าโสหุ้ยสูงหน่อย ถ้าเปรียบเทียบรหัสของ SMC กับ TKU รุ่น 3 ดูแล้วจะมีรหัสไม่ตรงกันเลยที่เป็นเช่นนี้เพราะ KTU เริ่มต้นที่ A1H แต่ของ SMC เริ่มที่ C0H ถ้าเป็นเช่นนี้จะเกิดอะไรขึ้น? สมมุติว่าผู้ใช้ที่มีระบบของ SMC หรือ TKU เช่นเดียวกันทุกอย่างคงจะไม่มีปัญหาอะไร แต่ถ้ามีผู้ใช้เครื่องไมโครคอมพิวเตอร์ระบบ TKU แต่ใช้เครื่องพิมพ์ของระบบ SMC ยกตัวอย่างเช่น เวลาใช้งาน CPU มีคำว่า "ฤษี" แล้วส่งให้เครื่องพิมพ์ เครื่องพิมพ์จะพิมพ์คำว่า "คฉี" ลักษณะคล้าย ๆ กับลงมหาฯ อะไรสักอย่างนะครับ (ไมโครฯ เราชักขลังขึ้นทุกที)

ข้อดีของระบบ SMC

รหัสแบบ SMC มีลักษณะการเรียงอักษรแบบสากล คือ ช่วงสัญญลักษณ์ช่วงตัวเลขรวมกับวรรณยุกต์ ช่วงพยัญชนะและช่วงสระเป็นลำดับไป ที่แตกต่างจากระบบ KTU คือระบบ SMC นำชุดไม้บังคับเสียงคือ ่, ้, ๊, ๋, ์ มานำหน้าชุดอักษรและหลังชุดอักษร และ จัด เรียงลำดับของสระแบบพจนานุกรมไทย ชุดนี้น่าเอ็นดูขึ้นมากสำหรับนักดูอักขระก็คือ ฦ ไม่ได้ถูกลืมไปแต่ถูกใส่ไว้ด้านหลัง ล. นอกจากนั้นระบบ SMC ยังนำชุด ่, ้, ๊, ๋, ์ มานำหน้าพยัญชนะและหลังชุดพยัญชนะเป็นชุดสระซึ่งเรียงลำดับกันตามแบบพจนานุกรมไทย การนำเอาชุดไม้ควบคุมเสียงไปไว้ด้านหน้าตัวพยัญชนะทำให้การสรรค์ (sort) ตามลำดับแบบพจนานุกรมไทยทำได้ง่ายขึ้นเพราะ ่, ้, ๊, ๋, ์ มีศักย์หรือลำดับการจัดอยู่เหนือพยัญชนะ ยกเว้นอักษรตัวแรกของคำที่จะต้องเป็นพยัญชนะสำหรับวรรณยุกต์นั้น ๆ ถึงแม้ว่าพยัญชนะตัวแรกจะมีสระนำหน้า เช่น เ, แ, โ ก็ถือว่ามีศักย์อยู่หลังพยัญชนะตัวแรกเสมอ ข้อดีอีกประการหนึ่งของการจัดชุดวรรณยุกต์ไว้ข้างหน้าพยัญชนะก็คือทำให้ชุดไม้คุมเสียงหรือวรรณยุกต์ซึ่งเป็นกลุ่มรหัสส่วนน้อย (minority element) ออกจากกลุ่มรหัสส่วนมาก (majority element) เสียได้ มิฉะนั้นแล้วก็จะต้องนำมาเรียงด้านหลังอักษรต่าง ๆ แบบ TKU ∅

(อ่านต่อฉบับหน้า)

วศิน เพิ่มทรัพย์

เปิดโลก คอมพิวเตอร์

ตอน

พูดจาภาษา BASIC

"บทความน่าอ่านสำหรับผู้เริ่มต้น"

"คุณรู้จักภาษา BASIC หรือยังครับ ?"

"แหม ไม่น่าถามเลย เล่นไมโครฯ แล้วไม่รู้จักภาษา BASIC ได้ยังไง"

"อือม์ ก็เคยได้ยิน เขาว่ามันใช้กับไมโครฯไม่ใช่หรือ?"

"รู้ แต่ไม่สน ผมเขียนแอสเซมบลีสบายใจกว่า"

"BASIC นี่มันอะไรนะครับ ? ฟังดูท่าทางจะง่าย ๆ ดี"

ครับ ก็แล้วแต่จะตอบกัน แต่อย่างหนึ่งที่ไม่มีใครปฏิเสธได้คือ ภาษา BASICเป็นภาษาคอมพิวเตอร์อีกภาษาหนึ่งที่ใช้กันแพร่หลายที่สุดในบรรดาภาษาอื่นอีก 4 ภาษาคือ FORTRAN;COBOL;BASIC และ Pascal ทั้งนี้เพราะความง่ายประการหนึ่ง กับมันเป็นภาษาที่ไมโครคอมพิวเตอร์ทุกเครื่องสามารถใช้ได้

ชื่อภาษา BASIC มาจากคำย่อของ Beginner's ALL-purpose Symbolic Insturction Code แปลว่าภาษาสารพัดประโยชน์สำหรับผู้เริ่มต้น คิดขึ้นโดย John G.Keneny และ Thomas Kurtz แห่งวิทยาลัย Dartmouth ที่บอกว่าภาษา BASIC แพร่หลายเพราะความง่ายนั้นก็เพราะนอกจากจะเป็นภาษาที่ใช้ง่ายแล้ว ตัวแปลภาษาที่ใช้ยังง่ายและมีขนาดเล็ก เหมาะสำหรับใช้ใช้งานในไมโครฯซึ่งมีหน่วยความจำขนาดเล็กอีกด้วย ถึงแม้ในปัจจุบันไมโครฯจะมีหน่วยความจำขนาดใหญ่ขึ้นมากก็ตาม แต่ก็ยังมีเครื่องเล็ก ๆ ราคาถูกอีกจำนวนมาก และภาษาก็ยังเป็นภาษาหลักของเครื่องเหล่านี้อยู่และก็คงจะเป็นเช่นนี้ไปอีกนาน

เวลาใช้งานไมโครฯนั้น ถ้าคุณเป็นผู้ใช้ประเภทที่เปิดเครื่องปุ๊บก็เรียกโปรแกรมสำเร็จรูปที่จ้างคนอื่นเขียนหรือซื้อ (หรือ COPY) เขามาใช้ ก็คงจะไม่ต้องเกี่ยวข้องกับภาษาคอมพิวเตอร์นัก แต่ถ้าเป็นนักเล่นไมโครฯประเภทที่ชอบทดลอง หรือชอบบงการให้มันทำงานอย่างโน้นอย่างนี้ตามใจคุณ ก็คงจะต้องใช้ภาษาคอมพิวเตอร์ โดยเฉพาะภาษา BASIC บ้าง ซึ่งก็ต้องใช้โปรแกรมสำเร็จรูปที่จะทำหน้าที่เป็นเสมือนล่ามคอยทำให้เจ้าไมโครฯมันเข้าใจภาษา BASIC ที่เราป้อนเข้าไป

ถ้าเป็นเช่นนั้นแล้ว เราก็น่าจะมาดูกันสักหน่อยว่า ภาษา BASIC ที่เราเห็นหน้าค่าตามันอยู่ทุกวันนี้นั้น แท้ที่จริงแล้วมันเป็นไงมาไงกันแน่

คอมไพเลอร์ (*เทียม*): ผู้บุกเบิกทางของ BASIC

ในระยะแรก ๆ ของการนำภาษา BASIC มาใช้กับไมโครพิวเตอร์นั้น ตัวแปลภาษาหรือโปรแกรมสำหรับแปลภาษา BASIC ที่ใช้จะอยู่ในรูปของคอมไพเลอร์ (compiler) ซึ่งจะทำหน้าที่แปลภาษาแต่เพียงอย่างเดียวเท่านั้น หมายความว่าในการใช้งานคอมไพเลอร์คุณจะต้องมีโปรแกรมที่เป็นภาษา BASIC เก็บอยู่ในเครื่องซึ่งเรียกว่า "Source program" แล้วเรียกคอมไพเลอร์มาแปลโปรแกรมนั้นจนได้เป็นโปรแกรมอีกอันหนึ่งที่เป็นภาษาเครื่อง (machine language) แล้วจึงจะใช้งานได้ ฟังดูก็ไม่ยากเท่าไหร่ แต่ลองนึกดูว่าถ้าคุณนั่งลงหน้าเครื่องและพยายามจะใช้มันจริง ๆ ปัญหาก็จะปรากฏขึ้นทันที อย่างเช่น คุณจะป้อนโปรแกรมภาษา BASIC เข้าไปในเครื่องได้อย่างไร? ตามปกติแล้วเมื่อคุณยังไม่ได้ใช้งานโปรแกรมใด ๆ เลย (เช่นเมื่อเปิดเครื่องขึ้นมาใหม่ ๆ) เครื่องจะทำงานอยู่ในส่วนของโปรแกรมควบคุมระบบ (operating system) ซึ่งก็จะรับเฉพาะคำสั่งที่มันรู้จักเท่านั้น คุณไม่อาจสั่งมันด้วยคำสั่งในภาษา BASIC ได้ จะป้อนโปรแกรมภาษา BASIC ให้มันเก็บไว้ก่อนก็ไม่ได้ ดังนั้นคุณจึงต้องมีโปรแกรมที่จะทำหน้าที่จัดเก็บโปรแกรมภาษา BASIC ที่คุณจะป้อนให้เข้าไปอยู่ในเครื่องตามลักษณะที่ถูกต้องเสียก่อน (อาจจะเอาไปเก็บลงหน่วยเก็บข้อมูล เช่น ดิสค์ก็ได้) โปรแกรมนี้ยังอาจมีความสามารถพิเศษที่จะช่วยคุณในการแก้ไข เพิ่มเติม ลบ หรือ ค้นหา ตัดต่อ ฯลฯ ข้อความหรือโปรแกรมที่คุณป้อนเข้าไปก็ได้ เราเรียกโปรแกรมนี้ว่า เอดิเตอร์ (editor) เป็นอันว่าคุณต้องมีโปรแกรมเอดิเตอร์นี้อีกตัวหนึ่งถึงจะป้อน source program ที่เป็นภาษา BASIC ได้

ทีนี้หันกลับมาดูที่คอมไพเลอร์กันใหม่ คอมไพเลอร์ภาษา BASIC สำหรับไมโครฯในยุคแรก ๆ นั้นเป็นแบบที่เราเรียกว่าคอมไพเลอร์เทียม (pseudo - compiler) คุณอาจสงสัยว่า *เอ๊ะมีเทียมมีแท้ด้วยหรือ?* คือมันเป็นอย่างนี้ครับ คอมไพเลอร์แท้นั้นจะแปล source program เป็นโปรแกรมภาษาเครื่องที่เรียกว่า object program ซึ่งตัวซีพียูหรือหน่วยประมวลผลกลางอันได้แก่ไมโครโปรเซสเซอร์ในเครื่องนั้นอ่านปุ๊บรู้ปั๊บเลยว่าจะให้มันทำอะไร วิธีแปลแบบนี้ยากหน่อย และโปรแกรมที่เป็นตัวแปลภาษาก็ต้องเก่งหน่อย แต่คอมไพเลอร์เทียมนี้ไม่ทำอย่างนั้น มันทำแค่แปล source program ให้อยู่ในรูปภาษาอีกภาษาหนึ่งที่คุณอ่านไม่รู้เรื่อง และตัวซีพียูก็อ่านไม่ออกด้วย ซึ่งเรียกว่า intermediate code อย่าเพิ่งงงครับว่าแปลทำไมถ้าไม่มีฝ่ายไหนอ่านออกเลย มีครับมี มี



โปรแกรมอีกอันหนึ่งซึ่งอ่านเจ้า intermediate code นี้ออก เราเรียกมันว่า อินเตอร์พรีทเตอร์ (interpreter) ซึ่งเจ้าโปรแกรมอันนี้แหละจะเป็นตัวที่ทำให้ซีพียูทำงานตามโปรแกรมที่เราต้องการได้ แต่มันทำได้อย่างไรผมจะยังไม่บอก ปล่อยให้คุณงงไปก่อน เอาเป็นว่าในการใช้งานนั้นคุณต้องมีเจ้าอินเตอร์พรีทเตอร์นี้อีกตัวหนึ่งก็แล้วกัน

สรุปแล้วกว่าจะทำงานได้ คุณต้องมี เอดิเตอร์, คอมไพเลอร์ (เทียม) และอินเตอร์พรีทเตอร์ และวิธีการใช้งานก็อาจลำดับได้ดังนี้คือ.....

- *เปิดเครื่อง เครื่องจะทำงานตามโปรแกรมควบคุมระบบทันที*
- *เรียกโปรแกรมเอดิเตอร์มาใช้*
- *ป้อนโปรแกรมภาษา BASIC เข้าไปเก็บไว้เป็น source program ในเครื่อง*
- *เลิกใช้โปรแกรมเอดิเตอร์ กลับไปใช้โปรแกรมควบคุมระบบ*
- *จากโปรแกรมควบคุมระบบ เรียกคอมไพเลอร์เทียมมาแปล source program เป็น intermediate code เก็บไว้ในเครื่อง*
- *เลิกใช้คอมไพเลอร์เทียม กลับไปใช้โปรแกรมควบคุมระบบ*
- *จากโปรแกรมควบคุมระบบ เรียกอินเตอร์พรีทเตอร์มาอ่าน intermediate code แล้วทำงานตามโปรแกรม*

ตามที่ผู้ผลิตซอฟท์แวร์หรือตัวแปลภาษา BASIC ทำมา ตัวคอมไพเลอร์เทียม กับอินเตอร์พรีทเตอร์จะทำมาคู่กัน จะเอาคอมไพเลอร์ของเจ้าหนึ่งไปใช้กับอินเตอร์พรีทเตอร์ของอีกเจ้าหนึ่งไม่ได้ มันจะไม่รู้เรื่องกันเพราะใช้ intermediate code ไม่เหมือนกัน แต่สำหรับโปรแกรมเอดิเตอร์แล้ว มักจะไม่ได้ทำมาด้วยกัน จะใช้ของบริษัท

ไหนก็ได้ขอให้เก็บ source program ตามรูปแบบ (format) มาตรฐานของเครื่องนั้น ๆ ก็พอแล้ว เพราะ source program จะถูกเก็บในรูปรหัส ASCII ซึ่งเป็นมาตรฐานสำหรับไมโครฯทั่วไป

ตัวอย่างของภาษา BASIC แบบที่กล่าวมานี้ก็คือ EBASIC หรือ BASIC-E ที่เขียนขึ้นโดย Gordon Eubanks Jr. อย่าแปลกใจที่คุณไม่เคยได้ยินชื่อ EBASIC เพราะในปัจจุบันนี้มันได้พัฒนามาเป็น CBASIC และ CBASIC 2 ส่วน CBASIC เป็นภาษาที่ใช้กันกว้างขวางพอสมควรในต่างประเทศ แต่ในเมืองไทยแล้วไม่ค่อยแพร่หลายนัก เพราะมีตัวแปลภาษาอีกแบบหนึ่งที่ใช้ง่ายกว่าเข้ามาแทน นั่นคืออินเตอร์พรีทเตอร์ภาษา BASIC ของบริษัทไมโครซอฟท์

อินเตอร์พรีทเตอร์: ใช้ง่ายสบายกว่า

ลองหันกลับไปดูขั้นตอนในการใช้งานภาษา BASIC แบบที่ใช้คอมไพเลอร์เทียม ทั้ง 7 ขั้นที่กล่าวมาแล้วดูซิครับ คุณดูให้ดีแล้วก็จะเห็นว่า ขั้นที่ 4 และ 6 นั้นไม่ได้ทำอะไรให้เป็นประโยชน์กับคุณเลย คุณต้องกลับสู่โปรแกรมควบคุมระบบก็เพราะว่าคุณไม่สามารถเรียกโปรแกรมอื่น ๆ มาใช้ได้ด้วยคำสั่งในโปรแกรมที่คุณใช้อยู่เดิม นอกจากนี้ในกรณีที่โปรแกรมมีที่ผิดและคุณต้องการแก้ไข คุณต้องกลับไปเรียกโปรแกรมเอดิเตอร์มาแก้ เรียกคอมไพเลอร์มาแปลใหม่ แล้วจึงเรียกอินเตอร์พรีทเตอร์มาทำงานตามโปรแกรมที่แก้แล้วอีกทีหนึ่ง ซึ่งไม่สะดวกนัก ดังนั้นจึงมีผู้คิดวิธีการใหม่โดยรวมเอาทั้งสามโปรแกรมเข้าด้วยกัน เมื่อคุณต้องการใช้งานคุณก็เรียกโปรแกรมชนิดนี้มาใช้เลย ซึ่งผมจะขอเรียกมันว่า "อินเตอร์พรีทเตอร์เบ็ดเสร็จ" เนื่องจากมันมีอินเตอร์พรีทเตอร์เป็นส่วนใหญ่ที่สุดและยังรวมเอาเอดิเตอร์และคอมไพเลอร์ (เทียม) ไว้ในตัวด้วย คุณจะสามารถป้อนโปรแกรมและให้เครื่องทำงานตามโปรแกรมนั้นได้ทันที หรือจะใช้งานกลับไปกลับมาอย่างระหว่างส่วนของเอดิเตอร์, คอมไพเลอร์ (เทียม) และ อินเตอร์พรีทเตอร์ เครื่องก็ยังคงทำงานอยู่ภายใต้โปรแกรมเดียวกันนั่นเอง ซึ่งสะดวกกว่าวิธีก่อนหน้านี้ โดยสรุปแล้วขั้นตอนในการใช้ภาษา BASIC ก็จะเหลือเพียง 2 ขั้น คือ

1. เปิดเครื่อง เครื่องจะทำงานตามโปรแกรมควบคุมระบบ
2. เรียกโปรแกรมอินเตอร์พรีทเตอร์เบ็ดเสร็จมาใช้งาน

ด้วยเหตุที่มันใช้ง่ายและอำนวยความสะดวกอย่างมากในการแก้ไขโปรแกรมนี่เอง ทำให้ตัวแปลภาษา BASIC แบบนี้ได้รับความนิยมสูงมากและแพร่หลายไปในหลายรูปแบบ ตัวอย่างเช่น บริษัทไมโครซอฟท์ได้ออกตัวแปลภาษา BASIC แบบนี้สำหรับเครื่องแบบต่าง ๆ จำนวนมากภายใต้ชื่อ MBASIC ซึ่งใช้กันกว้างขวางที่สุด และยังทำให้บริษัทผู้ผลิตคอมพิวเตอร์อื่น ๆ อีกหลายราย เช่นของญี่ปุ่นพัฒนาตัวแปลภาษา BASIC สำหรับเครื่องของตนโดยใช้ MBSIC เป็นต้นแบบเกิดเป็นภาษา BASIC ที่มีลักษณะคล้ายคลึงกันอีกมาก

อินเตอร์พรีทเตอร์ใน ROM

ถ้าคุณย้อนกลับไปดูขั้นตอนในการใช้งานที่เหลือเพียง 2 ขั้นของอินเตอร์พรีทเตอร์จะเห็นว่าถ้าเครื่องของคุณมีขนาดเล็ก และใช้ได้เฉพาะภาษา BASIC แล้วขั้นตอนที่ 1 และ 2 อาจรวมกันได้เป็นขั้นเดียวคือ

1. เปิดเครื่อง เครื่องจะทำงานตามโปรแกรมอินเตอร์พรีทเตอร์ทันที

ซึ่งวิธีนี้เป็นวิธีที่ใช้กันอยู่ในไมโครฯ จำนวนไม่น้อย โดยรวมเอาโปรแกรมควบคุมระบบเข้ากับอินเตอร์พรีทเตอร์เป็นโปรแกรมเดียวกัน และบรรจุในไอซีหน่วยความจำแบบถาวร หรือ ROM ดังนั้นเมื่อเปิดเครื่องขึ้นมา เครื่องจะทำงานตามโปรแกรมอินเตอร์พรีทเตอร์และพร้อมจะรับคำสั่งจากคุณได้ทันที ตัวแปลภาษาแบบนี้เราเรียกว่า **resident interpreter**

ข้อเสียประการหนึ่งของ resident interpreter ก็คือ เนื่องจากมันใช้เนื้อที่ในหน่วยความจำอย่างถาวรไม่สามารถจะเอาออกเมื่อไม่ใช้ภาษา BASIC หรือต้องการเนื้อที่ทำอย่างอื่น ดังนั้นมันจึงต้องถูกจำกัดไม่ให้มีขนาดใหญ่เกินไปในขณะใช้งาน ในหน่วยความจำจะต้องมีที่พอสำหรับเก็บโปรแกรมอินเตอร์พรีทเตอร์ โปรแกรมของคุณ ข้อมูลที่คุณจะใช้ และต้องมีที่เหลือสำหรับใช้ในการคำนวณหรือเก็บข้อมูลชั่วคราวที่จำเป็นในการทำงานของเครื่องอีกด้วย ดังนั้นจึงไม่สามารถทำให้ resident interpreter มีขนาดใหญ่และซับซ้อนหรือมีขีดความสามารถสูง ๆ ได้ ในคอมพิวเตอร์เครื่องที่มีหน่วยความจำน้อยมาก ๆ ตัวแปลภาษา BASIC ที่ใช้อาจจะเล็กจนเรียกได้ว่าเป็น tiny BASIC หรือ BASIC ขนาดจิ๋วคือมีคำสั่งต่าง ๆ ให้ใช้น้อย ทำงานได้น้อยอย่างนั่นเอง

ตัวอย่างของ resident interpreter เช่นในเครื่อง Apple II เป็นต้น ซึ่งถ้าไม่ได้ติดตั้งดิสค์ไดรฟ์ก็ใช้ได้กับเทปเพราะส่วนของโปรแกรมควบคุมระบบที่อยู่ใน ROM ทำไว้สำหรับใช้กับเทป แต่เมื่อติดตั้งดิสค์ก็ต้องเปลี่ยนโปรแกรม

ควบคุมระบบเสียใหม่โดยเรียกเอาโปรแกรมควบคุมระบบที่ใช้กับดิสค์ (Disk Operating System หรือ DOS) เข้ามาใช้แทน และเลิกใช้โปรแกรมควบคุมระบบเดิม แต่ส่วนที่เป็นอินเตอร์พรีทเตอร์ก็ยังคงใช้ที่อยู่ ROM ต่อไปตามเดิม

โครงสร้างภาษา BASIC กับการทำงานของอินเตอร์พรีทเตอร์

ถึงตรงนี้ก็จะได้อธิบายให้คุณผู้อ่านทราบถึงการทำงานของโปรแกรมอินเตอร์พรีทเตอร์ แบบที่ริเริ่มโดยบริษัทไมโครซอฟท์นี้เสียที แต่ก่อนอื่นก็ต้องขออธิบายถึงโครงสร้างของภาษา BASIC แบบนี้เสียก่อน

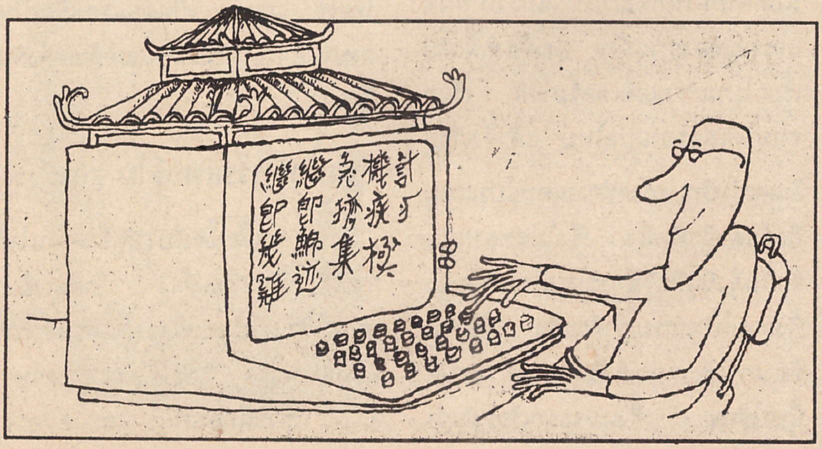
ภาษา BASIC แบบนี้เป็นแบบที่เราพบเห็นกันอยู่เสมอ ๆ นั่นเอง คือมีลักษณะการป้อนคำสั่งได้ 2 วิธี วิธีแรกเป็นวิธีลัดคือป้อนคำสั่งให้แก่ส่วนที่เป็นคอมไพเลอร์เทียม ซึ่งจะแปลเป็น intermediate code และส่งให้อินเตอร์พรีทเตอร์ทำตามเลย การป้อนคำสั่งวิธีนี้ จะป้อนคำสั่งโดยไม่มีตัวเลขบอกบรรทัด (line number) นำหน้า และเมื่อกดปุ่ม Return เพื่อบอกเครื่องว่าจบบรรทัด คำสั่งก็จะถูกทำตามโดยทันที

วิธีที่สองในการป้อนคำสั่งคือป้อนโปรแกรมโดยมี line number นำหน้า วิธีนี้เครื่องจะยังไม่ทำงานตามโปรแกรมที่เราป้อนเข้าไปทันที แต่จะเก็บโปรแกรมนั้นเอาไว้ และเมื่อเราป้อนโปรแกรมเข้าจนครบทุกบรรทัดแล้ว จะสั่งให้ทำงานตามโปรแกรมได้ด้วยคำสั่งต่างหากอีกทีหนึ่ง ซึ่งเป็นการเรียกอินเตอร์พรีทเตอร์มาทำงานโดยเริ่มทำจากบรรทัดที่มี line number แต่ก่อนจะข้ามไปถึงจุดนั้น เราลองมาดูกันก่อนดีกว่าว่าภายในเครื่องโปรแกรมถูกเก็บอย่างไร

ปกติแล้วโปรแกรมแต่ละบรรทัดจะถูกเก็บเรียงกันตามลำดับของ line number จากน้อยไปหามาก 10 อยู่ก่อน 100 100 อยู่ก่อน 1000 เป็นต้น ถ้าคุณป้อนโปรแกรมเข้าไปเรื่อย ๆ ตามลำดับของ line number ก็ไม่มีปัญหาอะไร แต่ถ้าคุณเกิดต้องการจะเพิ่มบรรทัดที่ 15 เข้าไประหว่างบรรทัดที่ 10 กับ 20 ที่ป้อนไปก่อนแล้วล่ะจะเป็นอย่างไร ?

วิธีการที่ใช้กันในการแก้ปัญหานี้ก็คือ จัดเก็บโปรแกรมแต่ละบรรทัดโดยมีข้อมูลเชื่อมโยงกันกล่าวคือ ในการป้อนโปรแกรม มันจะถูกเก็บเรียงกันตามลำดับที่ป้อน เช่น 10 แล้วตามด้วย 20, 30 (เรานิยมเว้น line number ให้กระโดดไปทีละ 10 หรือ 100 เพื่อให้มีตัวเลขที่จะใช้เป็น line number สำหรับบรรทัดที่ต้องการแทรกไว้ตรงกลาง ถ้าต้องมีขึ้นมา) ขณะเดียวกันในบรรทัดที่ 10 ก็จะมีข้อมูลเพิ่มเติม (ที่เครื่องเพิ่มให้เอง) บอกว่าบรรทัดต่อไปเลขที่เท่าไหร่ เช่นที่ 10 ก็บอกว่าบรรทัดถัดไปคือ 20 แต่เมื่อคุณป้อนบรรทัดที่ 15 เข้าไปหลัง 30 มันก็จะถูกเก็บอยู่หลัง 30 นั่นแหละ แต่ข้อมูลที่ว่าเมื่อครู่นี้ในบรรทัดที่ 10 จะถูกเปลี่ยนใหม่เป็นบอกว่าบรรทัดถัดไปคือ 15 และข้อมูลแบบเดียวกันในบรรทัดที่ 15 เองก็จะบอกว่าบรรทัดถัดไปคือ 20 ด้วยวิธีนี้เวลาที่อินเตอร์พรีทเตอร์อ่านโปรแกรมจึงสามารถทำงานตามลำดับคือ 10 15 และ 20 ได้ตามที่คุณต้องการ

เวลาที่คุณป้อนโปรแกรมเข้าไปโดยมี line number นำ เมื่อจบบรรทัดหนึ่งคุณก็กดปุ่ม Return เป็นการบอกให้เครื่องรู้ว่า จบบรรทัดหนึ่งแล้ว ขึ้นบรรทัดใหม่ ถ้าเป็นโปรแกรมเอดิเตอร์ธรรมดา (อย่างที่เราพูดถึงในตอนต้น) มันก็คงจะเอาโปรแกรมบรรทัดนั้นไปเก็บไว้เป็นรหัส ASC II เฉย ๆ ไม่มีอะไรต้องทำอีก แต่ในอินเตอร์พรีทเตอร์แบบนี้ ตลอดเวลาที่คุณคีย์โปรแกรมเข้าไปแต่ละคำสั่ง คอมไพเลอร์ภายในจะทำการแปลตามไปตลอดเวลา ดังนั้นเมื่อคุณกด Return แสดงว่าจบ 1 บรรทัด คุณก็จะได้ intermediate code 1 บรรทัดในทันที (บางบริษัทก็ใช้วิธีว่า เมื่อคุณกด Return แสดงว่าจบ 1 บรรทัด คอมไพเลอร์ก็จะถูกเรียกมาแปลโปรแกรมในบรรทัดนี้เสียทีหนึ่ง แบบนี้คุณอาจสังเกตได้เพราะเมื่อกด Return แล้วถ้าโปรแกรมบรรทัดนั้นยาว เครื่องจะเงียบหายไปวูบหนึ่งเพราะมันกำลังแปลอยู่ เมื่อแปลเสร็จแล้วจึงจะมารับ



คำสั่งต่อ)

Intermediate code ที่พูดถึงมาตั้งแต่แรกนั้น มันจะประกอบด้วยชิ้นย่อย ๆ หรือ token คำสั่งในภาษา BASIC แต่ละคำก็ถูกแปลเป็น token หนึ่งแทนที่จะเก็บเป็นคำเต็ม ๆ เพื่อความสะดวกและรวดเร็วในการทำงานของอินเตอร์พรีทเตอร์ และประหยัดที่ในหน่วยความจำ เช่น PRINT ถ้าเก็บเป็นตัวอักษร ต้องใช้ที่ถึง 5 ไบท์ อาจแปลเป็นรหัสตัวเลข เช่น 186 ซึ่งกินที่เพียง 1 ไบท์เท่านั้น ดังนั้นในแต่ละบรรทัดหรือประโยคของภาษา BASIC ก็จะถูกแปลเป็น token หลาย ๆ อันเรียงต่อกันไป

เมื่อป้อนโปรแกรมเสร็จแล้วก็เรียกอินเตอร์พรีทเตอร์มาทำงาน โดยมากมักใช้คำสั่ง RUN ซึ่งตรงนี้ที่เราได้ค้างกันไว้ในตอนต้นขณะที่พูดถึงอินเตอร์พรีทเตอร์แบบแรก ซึ่งผมจะเล่าถึงการทำงานของอินเตอร์พรีทเตอร์ไว้ตรงนี้เลยดังนี้ เมื่อคุณเรียกอินเตอร์พรีทเตอร์มา มันจะอ่าน intermediate code หรือ token แล้ววิเคราะห์ว่าจะต้องใช้ชุดคำสั่งใดภายในตัวอินเตอร์พรีทเตอร์เองมาทำงานเพื่อให้ได้ผลตามคำสั่งนี้ ในการอ่านนั้น แต่ละ token จะถูกมองแยกจากกัน แต่ละตัวก็มีความหมายเฉพาะอย่างต่ออินเตอร์พรีทเตอร์ เช่นคำสั่ง PRINT A + 1 แยกเป็น 4 token คือ

PRINT A + 1

เมื่ออินเตอร์พรีทเตอร์พบ PRINT มันก็จะรู้ว่าจะต้องพิมพ์สิ่งที่ตามมาทั้งหมด แต่เมื่ออ่านต่อพบ A ก็รู้ว่ามีค่าของตัวแปร A มาเกี่ยวข้อง อ่านพบ + ก็รู้ว่าต้องมีการบวก และพบเลข 1 ก็รู้ว่า A จะบวกกับ 1 ถ้าไม่มีอะไรต่อไปอีก ซึ่งเมื่ออ่านต่อก็พบว่าจบคำสั่งจริง จึงไปทำชุดคำสั่งภายในตัวที่ทำหน้าที่บวก โดยส่งค่าของตัวแปร A กับค่า 1 ไปเป็นข้อมูลให้ เมื่อได้คำตอบแล้วก็ไปทำชุดคำสั่งสำหรับพิมพ์ โดยส่งคำตอบที่ได้มานั้นไปให้เป็นข้อมูลอีกทีหนึ่งจะเห็นได้ว่าอินเตอร์พรีทเตอร์จะไม่แปล แต่จะอ่านคำสั่งจนจบประโยคและทำการวิเคราะห์ว่าจะต้องทำอะไรและทำโดยใช้ชุดคำสั่งส่วนไหนบ้างแล้วก็ทำไปตามนั้น

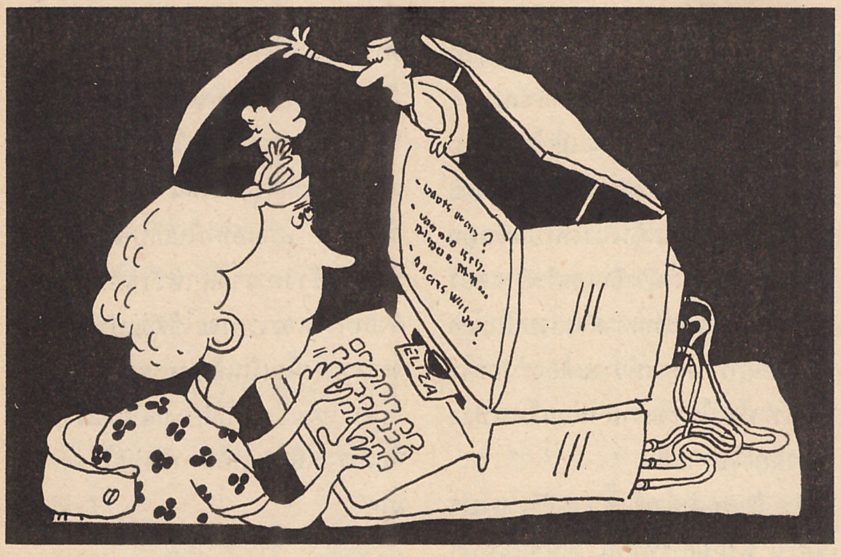
ถ้าคุณมีคำสั่งซ้ำ ๆ กันในโปรแกรม หรือมีการทำวน (LOOP) ให้ทำคำสั่งเดิม ซ้ำ ๆ กัน เช่น PRINT A +1 เมื่ออินเตอร์พรีทเตอร์ทำการวิเคราะห์และทำงานไปครั้งหนึ่ง จนกลับมาทำอีกมันก็ไม่สามารถจะจำได้ว่ามันเคยวิเคราะห์คำสั่งนี้แล้ว มันจำเป็นจะต้องทำการวิเคราะห์ใหม่อีกเที่ยวหนึ่ง พูดง่าย ๆ ก็คือวนมาเจอก็วิเคราะห์ไปเรื่อย ๆ ไม่สนใจว่าซ้ำหรือไม่ซ้ำเดิม ดังนั้นจุดนี้จึงเป็นเหตุให้การทำงานของอินเตอร์พรีทเตอร์ช้ากว่าคอมไพเลอร์ (แท้) ที่จะกล่าวถึงต่อไป

นอกจากนี้ในเรื่องการจัดที่ให้ตัวแปร ภาษา BASIC มีตัวแปร 2 ชนิดใหญ่ ๆ คือ **string: เก็บตัวอักษรหรือข้อความ** ซึ่งมีขนาดไม่คงที่ แล้วแต่ความยาวของข้อความ กับ **numeric: เก็บตัวเลข** ซึ่งมีขนาดคงที่ (อาจแบ่งเป็นเก็บเลขจำนวนเต็มหรือเลขที่มีทศนิยมขนาดต่าง ๆ ได้อีก แต่ก็มีขนาดคงที่เหมือนกัน) ในการทำงานของอินเตอร์พรีทเตอร์ จะใช้วิธีจัดที่ให้ตัวแปรเมื่อพบเป็นครั้งแรกในขณะที่ทำงานตามโปรแกรมดังนั้นอินเตอร์พรีทเตอร์จึงต้องสร้างตารางขึ้นภายในสำหรับเก็บว่า มีตัวแปรใดเคยพบมาแล้วบ้างและจัดที่ให้ที่ไหนถ้าเกิดไปเจอตัวแปรเข้าตัวหนึ่งในโปรแกรม จะได้ไปค้นหาในตารางได้ว่าเป็นตัวแปรเก่าหรือใหม่ ถ้าเป็นตัวแปรใหม่จะได้จัดที่ให้ใหม่ ถ้าเป็นตัวแปรเก่าก็จะได้รู้ว่าค่าของตัวแปรนั้นถูกเก็บอยู่ตรงไหน ถ้าจะใช้จะได้อ่านค่ามา หรือเอาค่าไปเก็บได้ถูก การที่ต้องสร้างตารางสำหรับเก็บข้อมูลดังกล่าว และเวลาจะใช้ตัวแปรก็ไปหาจากตารางนั้นเป็นอีกเหตุหนึ่งที่ทำให้ภาษา BASIC แบบนี้ทำงานได้ไม่เร็วมากนักเพราะเสียเวลาค้นหาข้อมูลในตารางมาก

คอมไพเลอร์(*แท้*): มาใหม่ทันใจกว่า

หลังจากที่อินเตอร์พรีทเตอร์ยึดครองตลาดของตัวแปรภาษา BASIC มานานก็ได้บังเกิดคอมไพเลอร์ที่เป็นคอมไพเลอร์จริง ๆ คือแปล source program ภาษา BASIC ให้เป็นภาษาเครื่องที่ซีพียูอ่านรู้เรื่องได้ทันที ผู้ผลิตคอมไพเลอร์เหล่านี้ออกมาก็ไม่ใช่ใครอื่น ก็คือ ผู้ผลิตอินเตอร์พรีทเตอร์อยู่แล้ว ซึ่งได้แก่บริษัทไมโครซอฟท์นั่นเอง ดังนั้นคอมไพเลอร์ภาษา BASIC ของไมโครซอฟท์นี้จึงไม่ได้มาแย่งตลาดของอินเตอร์พรีทเตอร์แต่ประการใดตรงกันข้ามกลับจะช่วยเพิ่มประสิทธิภาพการใช้งาน MBASIC ให้มากขึ้น ทั้งนี้เพราะโปรแกรมที่เขียนขึ้นตามกฎเกณฑ์ของ MBASIC จะสามารถแปลได้ด้วย BASCOM หรือคอมไพเลอร์ภาษา BASIC ของไมโครซอฟท์ซึ่งจะทำให้การทำงานของโปรแกรมรวดเร็วขึ้นมากทีเดียว

ส่วนทางด้าน CBASIC นั้น บริษัทดิจิตอลรีเสิร์ซ ผู้ผลิตโปรแกรมควบคุมระบบ CP/M ที่เป็นที่รู้จักกันดีก็ได้ออกคอมไพเลอร์สำหรับแปลภาษา BASIC ที่เขียนขึ้นตามกฎเกณฑ์ของ CBASIC ให้เป็นภาษาเครื่องภายใต้ชื่อ CB–80 โดย



แปลจาก intermediate code นั่นเอง ทั้ง BASCOM ของไมโครซอฟท์และ CB–80 ของดิจิตอลรีเสิร์ชต่างก็เป็นคอมไพเลอร์แท้ การทำงานของคอมไพเลอร์เหล่านี้อาจอธิบายได้ทำนองนี้คือ

สมมุติว่าผมใช้ตัวอย่างเดิมคือคำสั่ง PRINT A+1 ที่เราเคยอ้างถึงในตอนอธิบายเรื่องอินเตอร์พรีทเตอร์ ในการแปลคอมไพเลอร์จะอ่านคำสั่ง ทำการวิเคราะห์ และแปลเป็นภาษาเครื่อง ซึ่งมีลำดับการทำงานคล้าย ๆ อย่างนี้

1. เอาค่าตัวแปร A มา

2. เอาค่า 1 มา

3. บวกทั้งสองค่าเข้าด้วยกัน ได้คำตอบเก็บไว้

4. ส่งคำตอบที่ได้ไปออกที่จอ

การวิเคราะห์และแปลนี้จะทำเพียงเที่ยวเดียว และ object program หรือโปรแกรมภาษาเครื่องที่ได้จะสามารถนำไปป้อนให้ซีพียูทำตามได้ทันที ดังนั้นถึงแม้จะมีการทำงานซ้ำ ๆ กัน ก็จะเสียเวลาเพียงที่ใช้กับการทำงานจริง ๆ เท่านั้น คือพอเจอคำสั่งก็ทำตามไปได้เลยไม่ต้องวิเคราะห์คำสั่งใหม่ในแต่ละเที่ยวที่ทำ และถ้าคุณดูข้อที่ 1 จะเห็นว่าการที่จะเอาค่าตัวแปร A มาได้นั้นตำแหน่งของตัวแปร A จะต้องกำหนดแน่นอน ซึ่งจะทำให้ในขั้นตอนถัดจากการแปลที่เราเรียกว่าการ link หรือการเอาโปรแกรมมาต่อกัน ดังนั้นในการใช้คอมไพเลอร์ (แท้) ก็ยังมีอีก 2 ขั้น คือ การแปล (compile) กับการเชื่อมต่อโปรแกรมหรือการ link จึงจะได้ object program ที่สมบูรณ์

ถ้าคุณเริ่มงง ก็ขอให้ลองนึกดูช้า ๆ ว่าคอมไพเลอร์นั้นไม่สามารถจะแปลโปรแกรมเป็นภาษาเครื่องได้อย่างสมบูรณ์ทั้งหมด คำสั่งบางคำสั่ง เช่น SIN หรือฟังชันตรีโกณมิติอื่น จะถูกแปลเป็นแค่คำสั่งเรียกใช้โปรแกรมย่อยอื่น ๆ อีกทีหนึ่ง ซึ่งโปรแกรมย่อยที่ว่านี้จะถูกเก็บอยู่ต่างหากในห้องสมุดโปรแกรม (program library) ที่ผู้ผลิตคอมไพเลอร์เตรียมมาให้แล้วบนดิสค์ วิธีนี้จะทำให้คอมไพเลอร์ทำงานได้เร็วกว่าที่จะแปลออกมาทั้งหมด และเนื่องจากโปรแกรมที่ได้ยังไม่สมบูรณ์นี่เอง จึงยังไม่มีการกำหนดตำแหน่งแน่นอนของตัวแปรต่าง ๆ ว่าจะอยู่ที่ไหนในหน่วยความจำเมื่อโปรแกรมทำงาน ขั้นตอนที่จะเอาโปรแกรมย่อยที่ขาดไปมาต่อกับโปรแกรมที่ได้จากการแปล เพื่อให้ได้โปรแกรมที่สมบูรณ์ และจัดแจงกำหนดตำแหน่งที่แน่นอนของตัวแปรต่าง ๆ นั้นก็คือการ link นั่นเอง โปรแกรมที่ทำหน้าที่ link และกำหนดตำแหน่งต่าง ๆ ให้แน่นอนลงไปในกรณีนี้เราเรียกว่า link-loader และเนื่องจากทั้งคำสั่งต่าง ๆ และตำแหน่งของตัวแปรต่าง ๆ ถูกกำหนดแน่นอนแล้วตั้งแต่แรก ดังนั้นซีพียู จึงสามารถทำตามโปรแกรมดังกล่าวได้อย่างรวดเร็วกว่าวิธีการอื่น ๆ ที่กล่าวมาตั้งแต่ต้น

ถึงแม้วิธีการใช้คอมไพเลอร์จะทำให้การทำงานของโปรแกรมเร็วที่สุดก็ตาม ปัญหาที่เกิดขึ้นก็คล้าย ๆ กับกรณีของคอมไพเลอร์เทียมนั่นเอง คือถ้าคุณแก้ไขโปรแกรมก็ต้องกลับไปแปลใหม่ และ link ใหม่ ซึ่งเสียเวลาดังนั้นทางออกที่ดีก็คือ *เขียน, ทดสอบ และแก้โปรแกรม ด้วยอินเตอร์พรีทเตอร์ เสียก่อน* เมื่อได้โปรแกรมที่ถูกต้องใช้การได้แน่นอนแล้วก็จึงค่อยใช้คอมไพเลอร์ทำการแปล และ link เป็นโปรแกรมภาษาเครื่องเก็บไว้ใช้ต่อไป

ที่กล่าวมาแล้วทั้งหมดก็คงพอจะอธิบายได้โดยสังเขปถึง ภาษา BASIC ที่ใช้กับไมโครคอมพิวเตอร์ในปัจจุบัน แต่ถ้าจะดูกันจริง ๆ แล้ว ภาษา BASIC ที่เราพบเห็นกันมากก็มีอยู่แค่ 2 ตัวเท่านั้น คือ MBASIC หรืออินเตอร์พรีทเตอร์ของไมโครซอฟท์ และ Applesoft ซึ่งเป็นอินเตอร์พรีทเตอร์ใน ROM ของเครื่อง Apple II ซึ่งทั้งสองตัวก็เป็นอินเตอร์พรีทเตอร์เบ็ดเสร็จที่มีทั้งเอดิเตอร์และคอมไพเลอร์เทียมอยู่ในตัวเหมือนกัน ส่วน CBASIC และคอมไพเลอร์แท้นั้นใช้กันน้อยกว่า อย่างไรก็ตาม หวังว่าคุณที่อ่านบทความนี้ก็คงจะได้ความเข้าใจอะไรบ้างไม่มากก็น้อย พบกันใหม่โอกาสหน้า สวัสดีครับ

คู่มือการใช้งาน SuperCalc

พิสุทธิ สถาพรภูริศักดิ์

แบบฝึกหัด

การคำนวณในใบสั่งซื้อ

แบบฝึกหัดนี้จะแสดงถึงความสามารถของ Supercalc ในการที่จะเลือกค่าจากตารางอ้างอิงที่ตั้งไว้และใช้ค่านี้ในการเฉลยปัญหา เขาอาจจะใช้เทคนิคนี้กับการคำนวณอื่น ๆ ได้ เช่นหาค่าปริมาตรของเหล็กเส้นก่อสร้างจากขนาดที่ได้หรือการหาค่าประมาณของการก่อสร้างจากค่าประเมินผลในตาราง นอกจากนั้นในวงการแพทย์อาจจะให้ในการหาปริมาณเหมาะสมของจำนวนยาที่ฉีดเข้าคนไข้จากค่าที่ได้จากสภาวะรอบข้างของคนไข้เอง โดยเสาะหาจากตาราง empirical ที่ทำไว้นั้น ในแบบฝึกหัดยังแสดงการคำนวณจากเสกลที่ได้พิจารณาขอบข่ายจำกัดไว้ก่อน และเปลี่ยนค่าภายในชุดที่รวบรวมเปอร์เซ็นต์ส่วนลด ภาษีขายและแฟคเตอร์เพิ่มเติมอื่น ๆ

การคำนวณใบสั่งซื้อของลูกค้านี้ ใช้เทคนิคการคำนวณที่จำเพาะบางทีอาจไม่เหมาะสมกับบริษัทที่คุณทำงานอยู่ แต่การแสดงตัวอย่างที่มาตรฐานอาจเป็นแนวทางการปรับปรุงของส่วนงานได้ ในตัวอย่างที่แสดงในรูป 1 เป็นรายงานซึ่งมี 3 ส่วน คือส่วนบนซึ่งเป็นใบสั่งซื้อเอง (จากแถวที่ 1 ถึง 28) ส่วนกลางคือจากแถวที่ 29 เป็น Salesperson Commission Report หรือรายงานส่วนสมนาคุณผู้ขาย ส่วนล่างของรายงานคือตารางต่าง ๆ 3 ชุด คือตารางราคาสำหรับผลิตภัณฑ์กระดาษ ตารางราคาสำหรับผลิตภัณฑ์เครื่องแก้วและตารางส่วนลดสำหรับการซื้อเป็นจำนวนเงินทั้งสิ้นในระดับต่าง ๆ

สรุปปฏิบัติการที่ใช้

1. จัดตั้ง format หรือรูปแบบของแผ่นงาน
2. ใส่สูตรทางคณิตศาสตร์

	A	B	C	D	E	F	G
1	INVOICE NUMBER						
2	DATE :						
3	CUSTOMER NAME						
4	ADDRESS :						
5	CITY :						
6	STATE :						
7	ZIP CODE :						
8	SALESPERSON NO						
9							
10	QUANTITY	ITEM NO.	DESCRIPTION	UNIT COST	TOTAL COST		
11	----------	----------	----------	----------	----------		
12	95	131		1.58	150.10		
13	136	275		4.68	636.48		
14	12	100		9.23	110.76		
15	45	225		9.54	429.30		
16				.00	.00		
17				.00	.00		
18				.00	.00		
19				.00	.00		
20	==========	==========	==========	==========	==========		
21				FREIGHT :			
22				SUB TOTAL :	1326.64		
23			18	DISCOUNT :	-238.80		
24				NET :	1087.84		
25			5.4	SALES TAX :	58.74		
26					==============		
27				GRAND TOTAL :	1146.59		
28							
29	SALEPERSON COMMISSION REPORT						
30	----------------------------						
31	SALESPERSON NO	0					
32	INVOICE NUMBER	0					
33	COMMISSION :	152.18					
34							
35							
36	PRICING TABLE		PRICING TABLE		DISCOUNT TABLE		
37	FOR PAPER PROD	PRICE	GLASS WARE	PRICE	AMOUNT	PERCENT	
38							
39	0	0	0	0	0	0	
40	100	9.23	200	7.56	100	10	
41	125	8.55	225	9.54	200	12	
42	128	5.65	226	5.12	300	15	
43	129	2.96	230	6.32	500	18	
44	130	2.75	255	1.25			
45	131	1.58	275	4.68			
46	132	2.36	276	5.32			
47	133	0	280	0			
48							
49							
50							

รูปที่ 1 ตัวอย่างรายงานของใบสั่งซื้อ**Custommer invoice report**

SuperCalc

3. ป้องกันเซลจากการใส่สูตรซ้อน
4. ใส่ข้อมูลเพิ่ม
5. การรักษา (saving)
6. การพิมพ์ (printion)

Function ที่ใช้ : *SUM, MAX, MIN, LOOKUP*

Command **หรือคำสั่ง**

FORMAT เพื่อปรับความกว้างของคอลัมน์

FORMAT เพื่อจัดคำอธิบายตามแนวคอลัมน์ไปทางขวา

FORMAT เพื่อจัดคำอธิบายตามแนวแถวไปทางขวา

FORMAT ให้แสดงค่าเป็นดอลล่าร์และเซ็นต์ (คือตัวเลขที่จุดทศนิยม 2 ตำแหน่งนั่นเอง)

REPEAT TEXT เพื่อขีดเส้น

REPLICATE เพื่อลอกเลียนสูตร

PROTECT เพื่อป้องกันการใส่สูตรซ้ำ

นอกจากนั้นยังมี UNPROTECT GLOBAL INSERT, SAVE, OUTPUT

ต่อไปนี้เป็นคำอธิบายลำดับการปฏิบัติคือ...

1. การขจัดตั้งแบบฟอร์มของแผ่นงาน

แบบฟอร์มคร่าว ๆ ของแผ่นงานแสดงในรูปที่ 1, โดยกำหนดให้ทุกคอลัมน์กว้าง 14 อักษร ปกติแผ่นงาน Super-calc **ทุกคอลัมน์จะกว้าง 9 อักษร การที่ให้ทุกคอลัมน์**กว้าง 14 อักษร ก็เพื่อที่จะบรรจุป้าย (label) ต่าง ๆ หรือหัวข้อเรื่องได้พอดีนั่นเอง ในการปรับความกว้างของทุกคอลัมน์ ให้ใส่คำสั่งดังนี้

พิมพ์:

/F เริ่มคำสั่ง FORMAT หรือคำสั่งเกี่ยวกับรูปแบบของแผ่นงาน

G GLOBAL (ทั้งหมด) หรือทุกคอลัมน์

14 จำนวนของช่องว่างที่ต้องการให้คอลัมน์กว้าง

RETURN **ให้คำสั่งปฏิบัติการได้**

ในกรณีที่เห็นว่าทุกคอลัมน์ไม่เหมาะที่จะถูกกำหนดให้มีความกว้าง 14 อักษรเท่า ๆ กันหมดก็อาจจะมาพิจารณากันใหม่ให้เหมาะสม (กรุณาดูรูปที่ 1 ใหม่) จึงเห็นว่าคอลัมน์ A ควรกว้าง 15 , B เป็น 8 , C เป็น 25 , D เป็น 10 และ E เป็น 20 เมื่อเป็นเช่นนี้เราก็อาจใส่ลำดับคำสั่งใหม่ดังนี้

1. /F, C, A^ 15^ สำหรับคอลัมน์ A (เครื่องหมายจั่วโบสถ์แทน Carriage return หรือปัดแคร่) ให้กว้าง 15 ช่องอักษร
2. /F, C, B^ 8^ คำสั่งให้คอลัมน์ B มีความกว้าง 8ช่องอักษร
3. /F, C, C^ 25^ คำสั่งให้คอลัมน์ C มีความกว้าง 25 ช่องอักษร
4. /F, C, D^ 10^ คำสั่งให้คอลัมน์ D มีความกว้าง 10 ช่องอักษร
5. /F, C, E^ 10^ คำสั่งให้คอลัมน์ E มีความกว้าง 10 ช่องอักษร

ผลจะปรากฏดังในรูปที่ 2

ปกติโปรแกรม Super calc จะเปลี่ยนตำแหน่งของ Cursor ไปในทิศทางแบบเดิมกับที่เคลื่อนไปยังเซลก่อนหลังจากที่ผู้ใช้กด return ดังนั้นในบางกรณีที่จำเป็นต้องกด Return เวลาใส่สูตรการคำนวณซับซ้อนและต้องกด Return หลายครั้ง โปรแกรม Super calc อาจจะพาซื่อ ย้ายตำแหน่ง CURSOR ไปยังตำแหน่งอื่นใด ดังนั้นเพื่อกับการเปลี่ยนตำแหน่งของ CURSOR หลังการกด Return ให้แก้โดย**คำสั่งกำจัดเคลื่อนหน้า** (Nexteliminate) คำยอ่คือ N

การใช้หน้าที่ Nex-teliminate เพื่อกัน CURSOR ย้ายตำแหน่งให้พิมพ์

/G เริ่มคำสั่ง GLOBAL (ทั้งแผ่นหรือทั้งสิ้น)

N คำสั่ง Next-eliminates

คำสั่งดังกล่าวนี้ ไว้ทำก่อนที่จะใส่สูตรยาว ๆ

SuperCalc

2. การพิมพ์รายการหรือว่าหัว (Heading) ลงตามคอลัมน์หรือแถว

การพิมพ์ว่าหัว (Heading) หรือบางทีก็เรียกป้ายกำกับ (label) บ้างนั้น ก่อนที่จะใส่ข้อความลงเซลใด ๆ จะต้องใส่สัญญลักษณ์ (") หรือเครื่องหมายคำพูดก่อนเพื่อบอกโปรแกรมว่าต่อไปนี้เป็นอักขระ (text) ที่เป็นป้าย (label) ไม่ใช่เป็นสูตร formular

```
  |      A      || B ||          C          ||    D   ||   E  ||   F  |
 1|                   The Ocean Grass Co. Ltd.
 2|                   12 Sukumvit 15,BKK 10500
 3|
 4|INVOICE NUMBER:
 5|          DATE:
 6| CUSTOMER NAME:
 7|       ADDRESS:
 8|          CITY:
 9|         STATE:
10|      ZIP CODE:
11|SALESPERSON NO:
12|----------------------------------------------------------------------
13| Quantity    *Item No*       Description      * Unit cost.*TOTAL COST
14|----------------------------------------------------------------------
15|
16|
17|
18|
19|
20|
21|
22|
23|======================================================================
24|                                                  Freight :
25|                                                Sub total :
26|                                                 Discount :
27|                                                      Net :
28|                                                Sales tax :
29|                                                            ===========
30|                                              Grand total:
31|
32|Salesperson commission report
33|--------------------------
34| Salesman No. :
35|  Invoice No. :
36|   Commission :
37|======================================================================
38|Table 1             Table 2                                    Table 3
39|(Pricing)           (Pricing)                                  discount
40| Paper product  price     Glass ware product     price   amount*percent*
41|
42|             0      0                      0         0       0        0
43|           100   9.23                    200      9.56     100       10
44|           125   8.55                    225      8.54     200       12
45|           128   5.56                    226      7.12     300       15
46|           129   2.96                    230      6.32     500        8
47|           130   2.75                    255      6.25
48|           131   1.58                    275      5.68
49|           132   1.36                    276      5.32
50|           133   1.29                    280      5.11
```

รูปที่ 2 แผ่นงานที่เปลี่ยนแปลงความกว้างของช่องและเพิ่มหัวกระดาษ

วาง CURSOR ลงบนเซลที่ต้องการ จากนั้นจึงพิมพ์:

"	เตรียมเซลเพื่อรับอักขระ (text)
xxx	พิมพ์ข้อความที่ต้องการเข้าไป
RETURN	ให้โปรแกรมรับคำสั่งไปทำได้
E	Entry - format รับคำสั่งจะต้องบ่งช่วงของเซล
E24	เซลบนซ้ายของสี่เหลี่ยมในประสงค์ที่ต้องการ
:	ถึง
E28	เซลล่างขวาของสี่เหลี่ยมในประสงค์ที่ต้องการ
Return	(หรือ ^) รับคำสั่งชุดแรก
$	ให้แสดงค่าเป็น $
Return	รับคำสั่งไปปฏิบัติการ

เซล E30 ที่ต้องการให้แสดงค่าเป็น $ (ดอลล่าร์)

F	เริ่มคำสั่ง FORMAT
E	Entry - format คำสั่งที่ต้องบ่งช่วงของเซล
E 30	เซลที่จะ Format
Return	รับคำสั่งชุดแรก

3. การลากเส้นกั้นระดับ

ในกรณีที่ต้องการใส่เส้นเดี่ยวหรือเส้นคู่เพื่อเป็นเส้นที่ลากขีดตลอด ให้ใช้คำสั่งกระทำซ้ำข้อมูลที่เป็นเส้นประเดี่ยวหรือประคู่ ในการบอกเครื่องให้ทำซ้ำ ๆ ตามข้อมูล (เส้นเดี่ยวหรือเส้นคู่) นี้ ให้ใส่สัญญลักษณ์ (') แล้วพิมพ์ - หรือ = ตามด้วย Return เมื่อถึงเซลไหนที่ต้องการให้เส้นประนี้หยุด ก็ใส่คำสั่ง (') แล้วกด Return, เส้นประหลังเซลนั้นจะอันตรธานไปในพริบตา

หลังจากที่ขีดเส้นแล้ว ทำให้ใบส่งของเด่นขึ้นมาเป็น 5 ส่วนคือ:

SuperCalc

- ส่วนของข้อมูลสำหรับแผนกจัดการ
- ส่วนอธิบายรายการของสินค้า
- ส่วนผลรวมต่าง ๆ
- ส่วนรายงานรายได้สมนาคุณสำหรับผู้ขายและ
- ส่วนของตาราง 3 ชุดด้วยกัน

การตระเตรียมเพื่อป้อนคำสั่งลงแผ่นงาน

ขอทบทวนคำสั่งในโปรแกรม Super calc อีกหน่อยเพื่อกันงงนะครับ คำสั่งระดับแรกมีอยู่ด้วยกัน 18 ชนิด หรือหน้าที่ (function) คือ B, C, D, E, F, G, I, L, M, D, P, Q, R, S, T, U, W, Z ประกอบด้วยคำสั่ง 2 ระดับและคำสั่ง 3 ระดับ ส่วนคำสั่งระดับเดียวเช่น (=, ', ") จะไม่มีสลัด (/ slash) นำหน้าคำสั่ง คำสั่งระดับเดี่ยวหมายความว่าถ้าใส่สัญญลักษณ์ของคำสั่งแล้วก็ใส่วัตถุประสงค์ได้ทันที คำสั่ง 2 ระดับและ 3 ระดับ อยู่รวมกัน และก่อนจะใช้จะต้องมีสลัด (slash) นำหน้าก่อน ตัวอย่างของคำสั่ง 2 ระดับ เช่น G (lobal) จะต้องตามด้วยคำสั่งระดับ 2 หรือคำสั่งรองคือ F(ormular), N(ext), B(order), T(ab), R(ow), C(ol), M(an), A(uto) เพื่อใส่คำสั่งของหรือคำสั่งระดับ 2 แล้วจะต้องใส่ช่วงของเซลและกด return แต่ F ไม่จำเป็นต้องใส่ช่วงกด Return ก็ทำงานเช่น G, F ^ ส่วนคำสั่งรองตัวอื่นเช่น R(ow) จะต้องใส่วัตถุประสงค์ซึ่งเป็นช่วงของเซลลงไปด้วยจึงจะสมบูรณ์เช่น /G, R, C1:C13 (เครื่องหมายจั่วโบสถ์แทน R หรือปัดแคร่)

คำสั่ง 3 ระดับคือคำสั่งที่ต้องป้อนคำสั่งถึง 3 ครั้งจึงสิ้นสุดกระบวนดำเนินการ เช่น FORMAT เช่นเมื่อเลือก FORMAT โดยพิมพ์ F แล้วจะต้องเลือกหน้าที่ (function) อีก 4 อย่างคือ G(lobal), E(ntry), C(olumn) หรือ R(ow) คำสั่งระดับ 2 นี้เรียกว่าระดับบอกช่วงของคำสั่ง เช่นถ้าเลือก E (ntry) เราจะต้องพิมพ์ช่วงเซล C1 : C10 ^ (แล้วส่งงวดแรกก่อนด้วย Return) จากนั้นจึงเลือกวัตถุประสงค์เช่น TR (test right) แล้วจึงตามด้วย CR เป็นอันเสร็จพิธีเครื่องจะปฏิบัติตามคำสั่งทันที แล้วจากที่ปรับความเข้าใจกันแล้วขอให้ผู้ทำงานเลือกช่วงของเซลที่จะใช้ต่อไป

4. การปรับคำอธิบายชิดขวา (right - justify)

เพื่อเริ่มต้นปฏิบัติการสั่งชิดขวา , พิมพ์:

/F	เริ่มคำสั่งรูปแบบ
E	Entry - format เป็นการบ่งชนิดของช่วง เซลว่าเป็นแบบสี่เหลี่ยมผืนผ้า (ไม่ใช่แบบ Row หรือ Column)
A1	เซลมุมบนขวาของสี่เหลี่ยมในช่วงต้องการ
:	ถึง
A11	เซลสุดท้ายของมุมล่างซ้ายของสี่เหลี่ยม
Return	รับคำสั่งชุดแรก
TR	สั่งชิดขวา (text right)
Return	รับคำสั่งไปปฏิบัติการได้

ให้ทำเช่นนี้กับการสั่งชิดขวาทุกครั้ง

5. การสั่งให้แสดงค่าเป็นดอลล่าร์และเซ็นต์

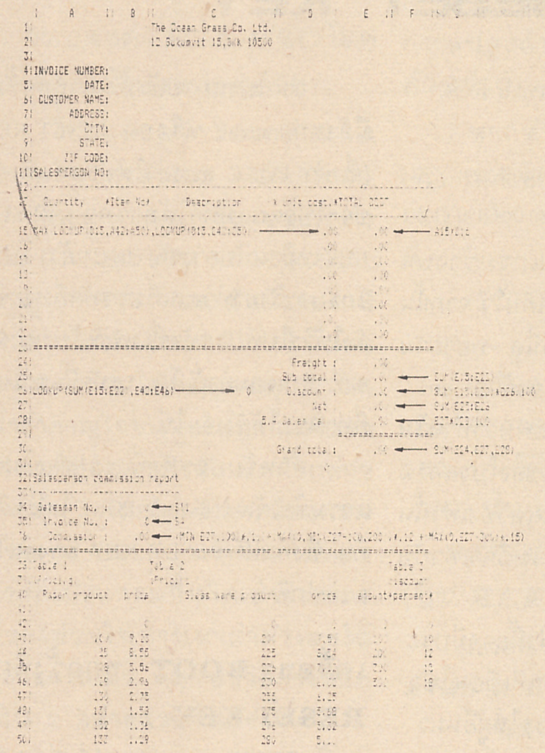
เพื่อให้ช่วงในคอลัมน์ที่บ่งจำนวนเงินแสดงผลเป็นดอลลาร์และเซ็นต์ นั่นก็คือทศนิยม 2 ตำแหน่งนั่นเอง (มิฉะนั้นโปรแกรมจะแสดงทศนิยมถึง 8 ตำแหน่งก็มี) เราจะใช้คำสั่งรูปแบบ (Format) เช่นเดิม และพิมพ์คำสั่งตามลำดับดังนี้

/F	เริ่มคำสั่ง Format จัดรูปแบบของแผ่นงาน
$	ให้แสดงค่าเป็นดอลล่าร์
Return	รับคำสั่งไปปฏิบัติ

คำชี้แจงเกี่ยวกับการใช้สูตรในแผ่นงาน

ก่อนที่จะใส่สูตร ซึ่งแสดงในรูปที่ 3 ขอแสดงอรรถาธิบายหน่อยนะครับ สูตรแรกที่จะต้องใส่ในเซล D 15 ตลอดถึง D 22 คือสูตรการหาค่ามากที่สุดของการตรวจค่าในตารางที่ 1 และ 2 จากซ้ายล่างเพื่อเปรียบเทียบกับค่า B(Item no) ในแถวเดียวกัน เมื่อได้ค่าแล้วให้นำค่ามากที่สุดมาเป็นค่า

unit cost ขอยกตัวอย่างเซล D 15 คือสูตร MAX (LOOKUP (B15, A42: A50) LOOKUP (B15,C42:C50)) ค่า total Cost ในเซล E 15 จะเป็นค่าของ Quantity (A15) คูณกับค่า D 15 ดังนั้นสูตรในเซล E 15 คือ AIS * DIS งานในลำดับต่อมาคือลอกเลียนสูตรลงมาตามแนวคอลัมน์ถึงแถวที่ 22



รูปที่ 3 แสดงสูตรประจำเซลต่าง ๆ

สูตรที่จะใส่ชุดที่ 2 คือสูตรของผลรวมต่าง ๆ ตั้งแต่เซล E25 ถึงเซล E29 เซล E25 เป็น Subtotal ใช้สูตร SUM (E15 : E22) ซึ่งเป็นผลรวมของ total cost ทั้งสิ้น เซล E26 เป็นการหาส่วนลดนั้นจะต้องหาเปอร์เซ็นต์ส่วนลดเสียก่อน, โดยกำหนดให้ C26 เป็นส่วนลดที่มีสูตร LOOKUP (SUM (E15 : E22) E42 : E46), สูตรนี้มีความหมายว่าให้รวมผลรวมตั้งแต่ E15 ถึง E22 เมื่อได้ค่าหนึ่งแล้วจึงนำไปตรวจดูค่าตั้งแต่ E42 ถึง E46 ว่ามีค่าเท่าใด ดังนั้นค่าส่วนลด E26 คือ Subtotal คูณกับส่วนลด ค่าของ E27 คือ Net ที่เป็นผลรวมของ E25 และ E26 ใช้สูตร SUM (E25 : E26) ส่วนภาษีขาย (Saletax) เท่ากับค่า Net คูณด้วย C 26/100 ค่า Grand total จะเท่ากับผลรวมของ Freight กับ Net และ Sale Tax เขียนเป็นสูตรได้ SUM (E24, E27, E28)

การใส่สูตรในช่วงที่ 3 คือใส่ในส่วนของ Saleperson commission report สูตรที่ใส่ใน saleperson no คือ BII สูตรที่ใส่ใน INVOICE number คือB4 และสูตรที่ใส่ใน B36 ของ commission คือ (MIN (E27, 100)*.10) + (MAX (D, MIN (E27-100,200)) *.12 (MAX +(O, E27-300) *.15) อธิบายได้คร่าว ๆ คือให้เลือกนำค่า 10 % ของค่าน้อยที่สุดระหว่าง E27 และ 100 หรือเลือกค่ามากที่สุดระหว่าง 0 กับค่าน้อยที่สุด 12 % ระหว่าง E27-100 และ 200 หรือเลือกค่ามากที่สุด 15% ระหว่าง D และ E27-30

∅



สนุกกับ apple II

ชูศักดิ์ วิจักขณา

การเปลี่ยนโปรแกรมทำงานของ RESET KEY

มีค่า POKE อยู่สองค่าที่สามารถเปลี่ยนการทำงานของ RESET KEY ให้ทำงานเหมือนกับการกดctrl-C ค่าที่เปลี่ยนนี้อยู่ในตำแหน่งของ DOS คือ-

- POKE 40286,35 และ -POKE -40287,216

หากท่านใส่ค่า POKE ข้างต้นนี้ไว้ในโปรแกรมของท่าน และ ใส่ ONERR GOTO. ไว้ในโปรแกรมเดียวกันนี้แล้ว ท่านสามารถสั่งให้เครื่องทำงานอะไรก็ได้ตามความต้องการของท่านทุกครั้งที่มีการกด RESET KEY หรือ ctrl-C โปรแกรมตัวอย่างข้างล่างนี้จะแสดงให้เห็นถึงการทำงานได้อย่างชัดเจน

```
10   ONERR GOTO 1000
15   POKE 40286,35: POKE 40287,-
     216
20   X = X + 1
30   PRINT X;"   ";:IF X < 400
     THEN 30
40   POKE 40286,60: Poke 40287,-
     212: END
1000 PRINT: PRINT "START CO-
     UNTING....";: X = 0: GOTO 30
```

ให้ท่านพิมพ์โปรแกรมข้างบนนี้ลงในเครื่อง แล้ว SAVE เก็บไว้ จากนั้นให้ท่านRUN โปรแกรมนี้ดู พยายามหยุดการทำงานของเครื่องด้วย ctrl-C และ RESET KEY ท่านจะพบว่าท่านไม่สามารถหยุดการทำงานของโปรแกรมนี้ได้เลย นอกจากจะปิดเครื่องเท่านั้น หรือค่า X ในโปรแกรมมีค่าเท่ากับ 400

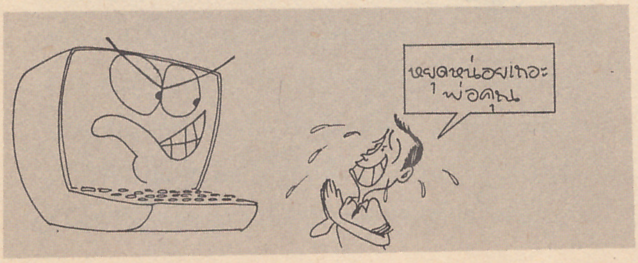
ค่า POKE สองค่านี้อาจมีประโยชน์สำหรับท่านผู้อ่านที่เขียนโปรแกรมเอง และไม่ต้องการให้ผู้อื่นมา -LIST ดูโปรแกรม หรือมาเปลี่ยนแปลงโปรแกรมของท่าน โดยใส่ NEW หรือ RUN หรือ DEL 0,999 หรือ PRINT CHR $ (4) "PR 6" หรือ PRINT CHR $ (4)"CATALOG" และอื่น ๆ ตามความต้องการของท่านไว้ในบรรทัดที่ 1000 นี้

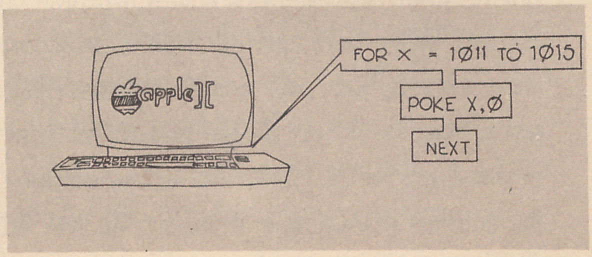
จากโปรแกรมข้างต้นนี้ ทุกครั้งที่มีการกด ctrl-C หรือกด RESET KEY (ซึ่งค่า POKE สองค่าในบรรทัดที่ 15 ของโปรแกรมทำให้การกด RESET KEY เหมือนกับการกด ctrl-C แล้ว) โปรแกรมจะวิ่งมาทำงานที่บรรทัดที่ 10 ซึ่งสั่งให้วิ่งไปที่บรรทัดที่ 1000 โปรแกรมจะทำงานตามคำสั่งในบรรทัดนี้ ตามแต่ที่ท่านจะได้สั่งเอาไว้

สำหรับบรรทัดที่ 40 ของโปรแกรมข้างต้น เป็นการ POKE ใส่ค่าเดิมคืน ซึ่งทำให้ RESET KEY ทำงานได้ตามปกติ

เครื่อง BOOT ทุกครั้งที่กด RESET KEY

มีค่า POKE อยู่ชุดหนึ่งซึ่งทำให้เครื่องคอมพิวเตอร์ APPLE II จะ BOOT ทุกครั้งที่มีการกด RESET KEY ให้ท่านนำข้อความข้างล่างนี้ใส่ไว้ในตอนต้นโปรแกรมของท่าน ขณะที่โปรแกรม RUN อยู่ หากท่านกด RESET KEY เครื่องจะ BOOT ใหม่ทันที





สำหรับโปรแกรมภาษาเบสิค ให้ใส่ข้อความดังนี้ไว้ตอนต้นของโปรแกรม

"FOR X = 1011 TO 1015 : POKE X,0 : NEXT"

สำหรับโปรแกรมที่ใช้เป็นภาษาเครื่องให้เปลี่ยนค่าตรงตำแหน่ง 3F3 ถึง 3F7ให้เป็นศูนย์ดังนี้-

3F3: 00 00 00 00 00

การแปลงค่า HEX-DEC โดยไม่ต้องใช้โปรแกรมช่วย

โปรแกรมที่จะช่วยท่านในการแปลงค่าตัวเลข HEXADECIMAL เป็น DECIMAL หรือแปลงค่าตัวเลข DECIMAL เป็น HEXADECIMAL มีอยู่มากมาย แต่เมื่อถึงคราวที่ท่านจะใช้มันขึ้นมาทีไร เป็นต้องหาโปรแกรมไม่ค่อยเจอทุกที มันน่ารำคาญใจจนพูดไม่ถูก ผมจึงขอเสนอวิธีแปลงค่าตัวเลขดังกล่าวข้างต้น โดยใช้เพียงเครื่องคอมพิวเตอร์ APPLE II เท่านั้น ไม่ต้องอาศัยโปรแกรมอื่นใดเข้าช่วยเลย ซึ่งจะช่วยให้ท่านหาค่าตัวเลขได้อย่างรวดเร็ว ผมแบ่งวิธีการออกเป็น 2 อย่างด้วยกันดังนี้-

1. การแปลงค่าจาก HEX.เป็น-DEC.

การแปลงค่าดังกล่าวนี้ให้ท่านทำตามลำดับต่อไปนี้

- พิมพ์ CALL-151 **แล้ว กด -** RETURN.

- พิมพ์ 75 :(LOW BYTE) (HIGH BYTE) NED 20G **แล้วกด** RETURN.

- เครื่อง APPLE II **จะพิมพ์ค่าตัวเลข** DEC. **ออกมาให้ท่านทันที**

คำอธิบาย

(LOW BYTE) **คือ ค่า** LOW BYTE **ของตัวเลข** HEX. **ที่ท่านต้องการแปลงค่าเป็น** DEC.

(HIGH BYTE) **คือ ค่า** HIGH BYTE **ของตัวเลข** HEX **ที่ท่านต้องการแปลงค่าเป็น** DEC.

ตัวอย่าง

หากท่านต้องการแปลงค่า HEX. **ของตัวเลข** FDED**ให้เป็นค่า -** DEC. **แล้ว ท่านก็เพียงแต่พิมพ์ดังนี้-**

75:ED FD NED 20G **แล้ว กด** RETURN.

เครื่อง APPLE II **ก็จะพิมพ์ตัวเลข 65005 ออกมา ซึ่งตัวเลข 65005 นี้ก็คือค่าตัวเลข** DEC. **ของ** FDED **นั่นเอง**

2. การแปลงค่าจาก DEC. เป็น HEX.

การแปลงค่าตามวิธีนี้ให้ท่านทำตามลำดับดังต่อไปนี้

- พิมพ์ NEW **แล้วกด** RETURN

พิมพ์ตัวเลขที่ท่านต้องการแปลงค่าลงไปในเครื่อง ตามด้วยตัวอักษร (ระหว่าง A **ถึง** Z **) ตัวใดตัวหนึ่ง แล้วกด** RETURN

จากนั้นให้ท่านพิมพ์ CALL--151 **แล้ว กด** RETURN

- พิมพ์ 803.804 แล้ว กด RETURN

- เครื่อง APPLE II **ก็จะพิมพ์ค่า** HEX. **ออกมาดังนี้**

0803 - (LOW BYTE) (HIGH BYTE) **ซึ่งเป็นค่า** LOW BYTE **และ** HIGH BYTE **ของตัวเลขที่ท่านต้องการแปลงค่าเป็น** HEX. **นั่นเอง**

ตัวอย่าง

เช่นท่านต้องการทราบว่าตัวเลข 60704 เป็นค่า HEX. เท่าใดท่านก็ต้องทำดังนี้

- พิมพ์ NEW แล้ว กด RETURN

- พิพม์ 60704 K แล้ว กด RETURN (K คือตัวอักษร ซึ่งท่านจะใช้เป็นตัวอักษรอื่นใดก็ได้)

- พิมพ์ CALL-151 แล้ว กด RETURN

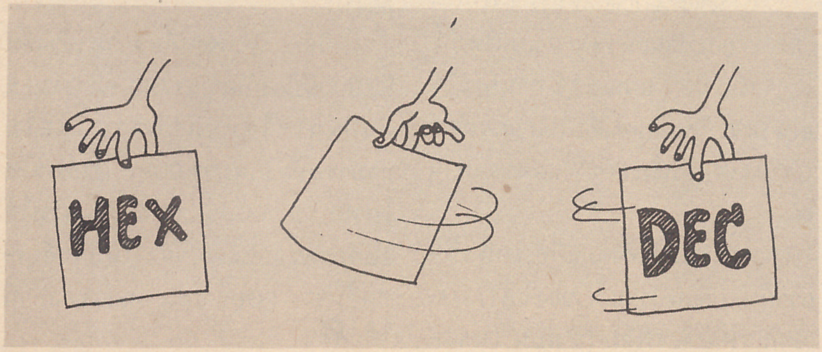
- พิพม์ 803.804 แล้ว กด RETURN.

- เครื่อง APPLE II. จะพิมพ์ค่า - HEX. ออกมาดังนี้0803-20 ED

ซึ่งจะบอกให้ท่านได้ทราบว่าตัวเลข 60704 แปลงเป็นตัวเลข HEX.จะมีค่าเท่ากับ ED20 นั่นเอง (โดยท่านนำตัวเลข LOW BYTE-20 มาต่อท้ายตัวเลข HIGH BYTE-ED ก็จะได้ค่า HEX. ออกมา) สำหรับการแปลงค่าที่มีเครื่องหมายติดลบอยู่ข้างหน้า ให้ท่านนำตัวเลขติดลบนั้นมาบวกเข้ากับตัวเลข 65536 ก่อน ได้ผลลัพท์เท่าไร จึงนำผลลัพท์ที่ได้มานี้ไปแปลงค่าตามวิธีดังกล่าวข้างต้น เช่น ท่านต้องการหาค่า HEX. ของตัวเลข-16369 ก็ให้ท่านนำไปบวกเข้ากับตัวเลข 65536 ดังนี้

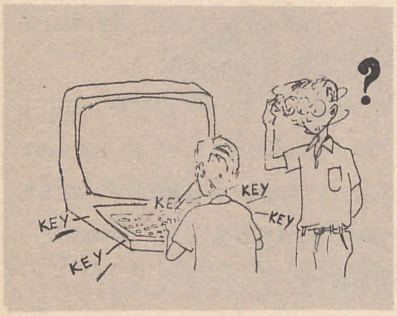
-16369 + 65536 = 49168

ได้ผลลัพท์เท่ากับ 49168 จากนั้นให้ท่านนำตัวเลข 49168 ไปดำเนินตามวิธีแปลงค่าที่กล่าวมาข้างต้นนี้ต่อไป



CATALOG และ LISTล่องหน

ท่านสามารถทำให้ CATALOG และโปรแกรมของท่านเมื่อ LIST ดูล่องหนได้โดยก่อนที่ท่านจะ CATALOG หรือ LIST ให้ท่านพิมพ์ดังนี้



POKE 50,128 แล้ว กด RETURN

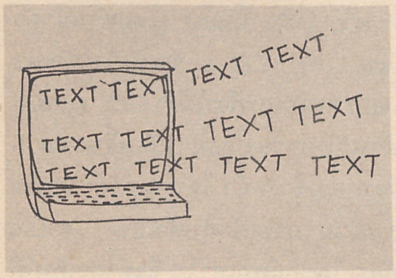
จากนั้นให้ท่าน CATALOG หรือ LIST โปรแกรม ท่านจะพบว่าท่านไม่สามารถมองเห็น CATALOG หรือโปรแกรมของท่านได้เลย

ถ้าต้องการให้เครื่องกลับเป็นปรกติให้ท่านพิมพ์ดังนี้

POKE 50,255 แล้ว กด RETURN

การ CATALOG และ LIST ก็จะสามารถมองเห็นได้ตามปรกติ

การ SAVEข้อความที่ปรากฏบนจอภาพ



ท่านสามารถ SAVE เก็บข้อความที่ปรากฏอยู่บนจอภาพ(TEXT SCREEN) ได้ โดยเพียงแต่พิมพ์ข้อความดังนี้

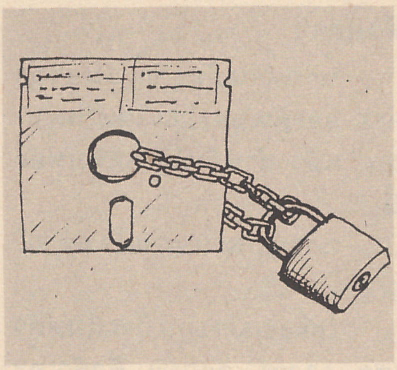
BSAVE TEXT SCREEN,A$400, L$3FF

แล้วกด RETURN เครื่องก็จะ SAVE ข้อความที่ปรากฏอยู่บนจอภาพในขณะนั้นเก็บลงไว้บนแผ่น DISK เมื่อใดที่ท่านต้องการข้อความนั้นกลับคืนมาปรากฏบนจอภาพอีกครั้งหนึ่ง ก็ให้ท่านพิมพ์ข้อความดังนี้

BLOAD TEXT SCREEN แล้ว กด RETURN

ข้อความเดิมนั้นก็จะกลับมาปรากฏขึ้นบนจอภาพเหมือนก่อนที่ท่านจะ SAVE เก็บทุกประการ

การป้องกันโปรแกรมบนแผ่น DISK ของท่าน



ท่านผู้อ่านทุก ๆ คน คงต้องการทราบวิธีป้องกันไม่ให้ผู้อื่นมา COPY แผ่น DISK ของท่านได้ง่าย ๆ ผมจึงขอเสนอวิธีป้องกันการ COPY แผ่น DISK อย่างง่าย ๆ ดังนี้

ขั้นที่ 1.การทำแผ่น DISKใหม่

A. ให้ท่านพิมพ์ BOOT แผ่น DISK ปรกติ (DOS 3.3)

B. ให้ท่านพิมพ์ POKE 44033, 16 แล้ว กด RETURN

C. นำแผ่น DISK ใหม่ใส่เข้าไปใน-DRIVE ของท่านแล้ว INIT แผ่นใหม่นี้

เมื่อ INIT แผ่นเสร็จ ท่านก็จะได้แผ่น DISK ใหม่ที่บุคคลอื่นไม่สามารถจะ-CATALOG ดูหรือ COPY แผ่นของท่านได้

การกระทำทั้งหมดที่ทำมานี้ เป็นการย้ายเปลี่ยน TRACK ของ CATALOG จาก TRACK 17เดิม (ซึ่งเป็น TRACK ปรกติ) มาอยู่ที่ TRACK 16 โดยการ POKE เปลี่ยนค่าที่ตำแหน่ง 44033 ก่อนที่จะ INIT แผ่นนั้นเอง

นอกจาก TRACK 16 ตามตัวอย่างข้างต้นนี้แล้ว ท่านอาจจะใช้ค่าอื่นระหว่าง 3-15 ในการ POKE เปลี่ยน TRACK ได้ โดยให้ท่าน POKEค่าตัวเลขนั้นที่ตำแหน่ง 44033

ขั้นที่ 2.การย้ายโปรแกรมจากแผ่น DISKปรกติมาเก็บลงบนแผ่น DISKใหม่

1. ให้ท่าน BOOT แผ่น DISK ปรกติ (DOS 3.3)
2. LOAD โปรแกรมจากแผ่น DISK ปรกติเข้ามาในเครื่อง
3. พิมพ์ POKE 44033,16 (สมมุติว่าท่านใช้ค่า 16) แล้วกด RETURN
4. นำแผ่น DISK ใหม่ซึ่งท่านได้ INIT ตามขั้นที่ 1. ข้างต้นนี้ ใส่เข้าไปใน-DRIVE ของท่าน
5. SAVE โปรแกรมที่มีอยู่ในเครื่อง เก็บลงไว้บนแผ่น DISK ใหม่นี้
6. พิมพ์ POKE 44033,17 (ค่าปรกติ) แล้วกด RETURN
7. ใส่แผ่น DISK ปรกติเข้าไปใน DRIVE แล้วให้ท่านดำเนินวิธีการตามข้อ2. ข้างต้น จนท่าน SAVE โปร-แกรมทั้งหมดลงบนแผ่น DISK ใหม่ของท่านครบถ้วนหมดแล้ว

เมื่อเสร็จเรียบร้อยแล้ว ท่านก็จะได้แผ่น DISK ใหม่ซึ่งบุคคลอื่นไม่สามารถจะ COPY โปรแกรมได้ง่าย ๆ อีกต่อไป

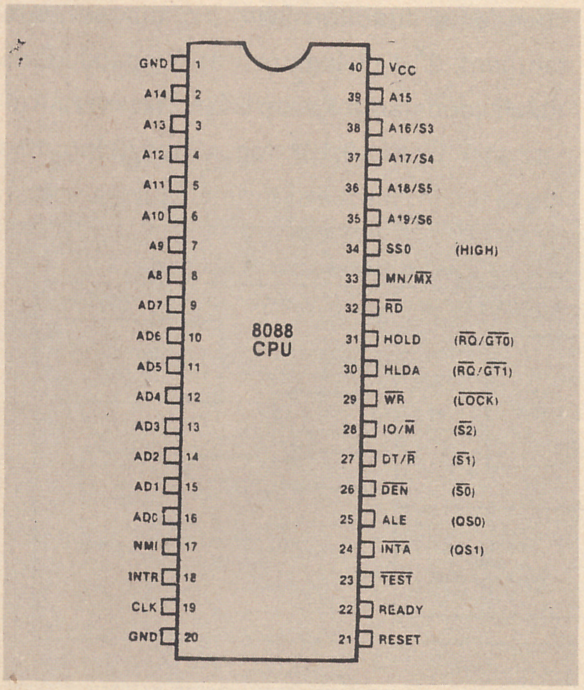
วิธีการทั้งหมดที่กล่าวมาข้างต้นนี้ เป็นวิธีป้องกันการ COPY อย่างง่าย ๆ ซึ่งมีผลใช้ได้พอสมควร แต่ไม่สามารถป้องกันการ COPY โดยการใช้โปรแกรม COPY บางโปรแกรมได้ แต่ก็อาจช่วยชะลอเวลา ในการที่บุค-คลอื่นจะ COPY แผ่น DISK ของท่านได้ ∅

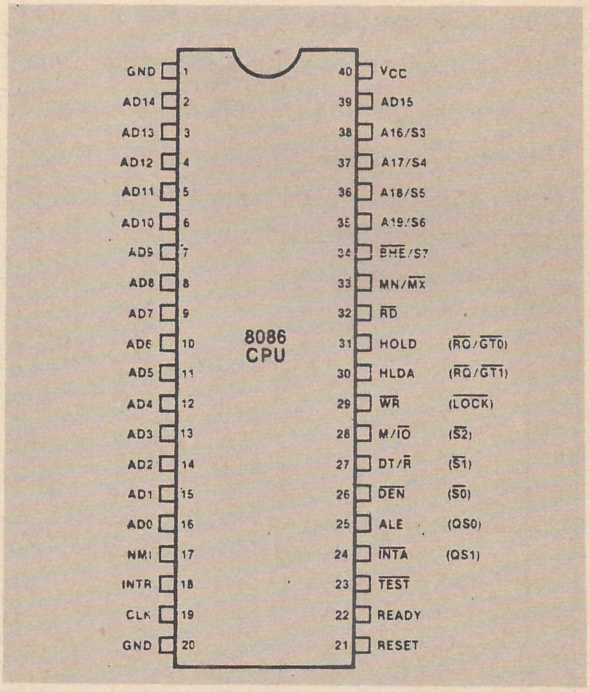
8086/8088

โดย ธวัช ศรีประดิษฐวรกุล

บทนำ

8086 และ 8088 เป็นไมโครโปรเซสเซอร์ ที่ผลิตโดยบริษัท อินเทล คอร์เปอร์เรชั่น โดยเป็นไมโครโปรเซสเซอร์ที่มีลักษณะโครงสร้างภายในเป็นแบบ 16 บิท คือ **Register** ทุกตัวมีขนาด **16 bit** มี **ALU** ขนาด **16 bit** และมีทางเดินของข้อมูลที่ติดต่อภายใน **(internal data path)** ขนาด **16 bit** 8088 ถูกออกแบบมาให้มีทางเดินของข้อมูลที่ติดต่อกับภายนอก **(external data path)** ที่จะติดต่อกับ หน่วยความจำ และ อุปกรณ์อินพุท–เอาท์พุท **(I/O)** เป็นแบบ **8 bit** ในขณะที่ 8086 มี **external data path** เป็นแบบ **16 bit** ทำให้สามารถเคลื่อนย้ายข้อมูลได้ทีละ 16 บิท 8086 และ 8088 โดยทั่วไปจะมีโครงสร้างภายในที่คล้ายกันมาก และสามารถใช้คำสั่งชุดเดียวกัน ทั้ง 8086 และ 8088 ถูกผลิตโดยใช้ **N-Channel, depletion Load, Silicon gate technology (HMOS)** และถูกบรรจุอยู่ในชิพ ไอ.ซี.ขนาด 40 ขา และสามารถใช้งานได้โดยมีแหล่งจ่ายไฟ เพียงระดับเดียวคือ 5 โวลท์ และสามารถทำงานได้ตามสัญญาณนาฬิกาขนาดสูงสุด **5 MHz** ลักษณะขาของ 8086 และ 8088 แสดงได้ดังรูปที่ 1





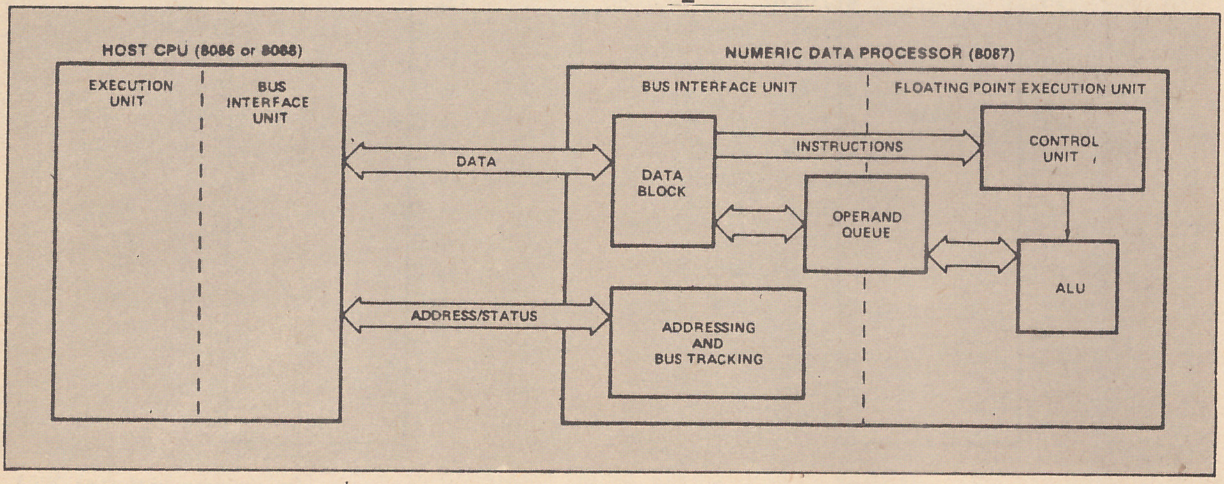
รูปที่ 1 การจัดขาของ 8086/8088

(MAXIMUM MODE PIN FUNCTIONS (e.g.. LOCK) ARE SHOWN IN PARENTHESES.)

8086,8088 ถูกออกแบบมาให้สามารถนำมาใช้งานได้อย่างกว้างขวางในระบบไมโครคอมพิวเตอร์สามารถนำมาออกแบบใช้งานตั้งแต่ระบบไมโครคอมพิวเตอร์ที่มีโปรเซสเซอร์เพียงตัวเดียว (single processor) และมี ไอ.ซี.ที่ประกอบเป็นระบบเพียงไม่กี่ตัวจนถึง Multiprocessor system คือมีโปรเซสเซอร์หลายตัวและยังสามารถมีหน่วยความจำได้ถึง 1 เมกกะไบท์ โดยปกติแล้ว 8086,8088 สามารถทำงานได้ 2 operation mode คือ Maximum mode และ Minimum mode การที่ 8086,8088 จะทำงานใน mode ใดนั้นสามารถทำได้โดยให้สัญญาณ ลอจิก แก่ขา Mn/$\overline{\text{Mx}}$ ของ CPU ซึ่งขา Mn/$\overline{\text{Mx}}$ นี้จะทำหน้าที่ define ลักษณะ Configuration ของระบบและกำหนดหน้าที่ของขา control ต่าง ๆ ถ้าขา Mn/$\overline{\text{Mx}}$ เป็น "1" CPU จะทำงานอยู่ใน Minimum mode /สัญญาณของขา control (ขา 21-34) ต่าง ๆ จะมี function อย่างหนึ่ง และสามารถให้สัญญาณควบคุมบัส (bus control signal) แก่ memory และ I/O ได้โดยตรง แต่ถ้าขา Mn/$\overline{\text{Mx}}$ เป็น "0" จะทำงานใน Maximum mode ซึ่งขา Control จะมี function ไม่เหมือนกับใน Minimum mode ซึ่งสัญญาณจากขา Control เหล่านี้จะใช้การควบคุมระบบขนาดกลางและใหญ่แทน โดยเฉพาะอย่างยิ่งในระบบ Multiprocessing System โดยจะมีอุปกรณ์สนับสนุน (Support Chip) 8288 ซึ่งผลิตโดยบริษัทเดียวกัน ทำหน้าที่เป็น Bus Controller ซึ่งจะทำงานตามสัญญาณควบคุมจาก CPU สัญญาณจากขาต่าง ๆ ซึ่ง CPU ถูก Program ใน Maximum mode สามารถนำมาถอดรหัสเพื่อผลิตสัญญาณควบคุมต่าง ๆ ให้แก่ส่วนที่เหลือของระบบลักษณะของ bus และ สัญญาณควบคุมจะเข้ากันได้ (Compatible) กับระบบ Multibus ของบริษัท INTEL จากลักษณะข้างต้นเราจะเห็นได้ว่า 8086, 8088 นั้น ถูกออกแบบเพื่อให้มีอุปกรณ์สนับสนุนทางฮาร์ดแวร์อยู่ภายในตัวชิพเลย (hardware support) ซึ่งสามารถนำมา inplement กับระบบ Multiprocessor ได้เป็นอย่างดี

CPU 8086 และ 8088 ถูกออกแบบมาให้สามารถใช้งานร่วมกับ 8089 ซึ่งเป็น Input/Output Processor (IOP) และ 8087 ซึ่งเป็น Numeric Data Processor (NDP) 8087 และ 8089 เป็น Co-processor ซึ่งถูกออกแบบมาเพื่อเพิ่มประสิทธิภาพแก่ระบบ 8087 มักจะใช้ในงาน (Application) ซึ่งต้องมีการคำนวณมาก ภายใน 8087 มี Register ซึ่งสามารถคำนวณ 80-bit floating point number ได้

8087 เมื่อถูกใช้งานเป็น Coprocessor ในระบบ Multiprocesser System เราสามารถใช้คำสั่งของ 8087 ในการทำ Arithmatic operation บน data type ต่าง ๆ หลายชนิด คือ ตัวเลข integer ขนาด 16, 32 และ 64 bit ตัวเลข floating point ขนาด 32, 64 และ 80 bit และตัวเลข BCD (Binary coded to Decimal) แบบ Pack จำนวนถึง 18 digit Arithamatic operation ที่ 8087 ทำได้ ได้แก่ บวก ลบ คูณ หารหา Square Root หาค่าสัมบูรณ์ และทำพวก Transcendental function ได้แก่ การ tangent, Arctantgent, 2^X-1, Y $\log_2(x+1)$, $Y\log_2(x)$ เป็นต้น ค่าคงที่ (constant) ที่สามารถเรียกใช้โดยตรงจากคำสั่งของ 8087 ได้แก่ค่า 0,1,π ,$\log_{10}2$,$\log_e 2$,$\log_2 10$ และ $\log_2 e$ จากการวัดประสิทธิภาพการใช้งานของ 8087 ใน



รูปที่ 2 โครงสร้างภายในของ 8087 Numeric Data Processor

งาน application ต่าง ๆ ที่ใช้การคำนวณมาเปรียบเทียบกับวิธีการเขียน Program เพื่อคำนวณหาค่าซึ่งต้องใช้ความรู้ทาง Numerical Analysis ในการแก้ปัญหาดังกล่าวแล้วปรากฏว่าการใช้ 8087 เป็น Numerical data processor สามารถทำงานคำนวณได้เร็วกว่าถึงประมาณ 100 เท่า สำหรับ 8089 ภายในตัวมันมีสอง I/O Channel ซึ่งเป็นอิสระต่อกัน และมี DMA Controller ภายในตัวของมันเองด้วยซึ่งเมื่อใช้ 8089 ร่วมกับ 8086 หรือ 8088 แล้วจะทำให้เราสามารถนำไปใช้งานในระบบประมวลผลข้อมูลซึ่งต้องมีอินพุทหรือเอาท์พุทมาก (I/O intensive Data Processing System) ได้เป็นอย่างดี

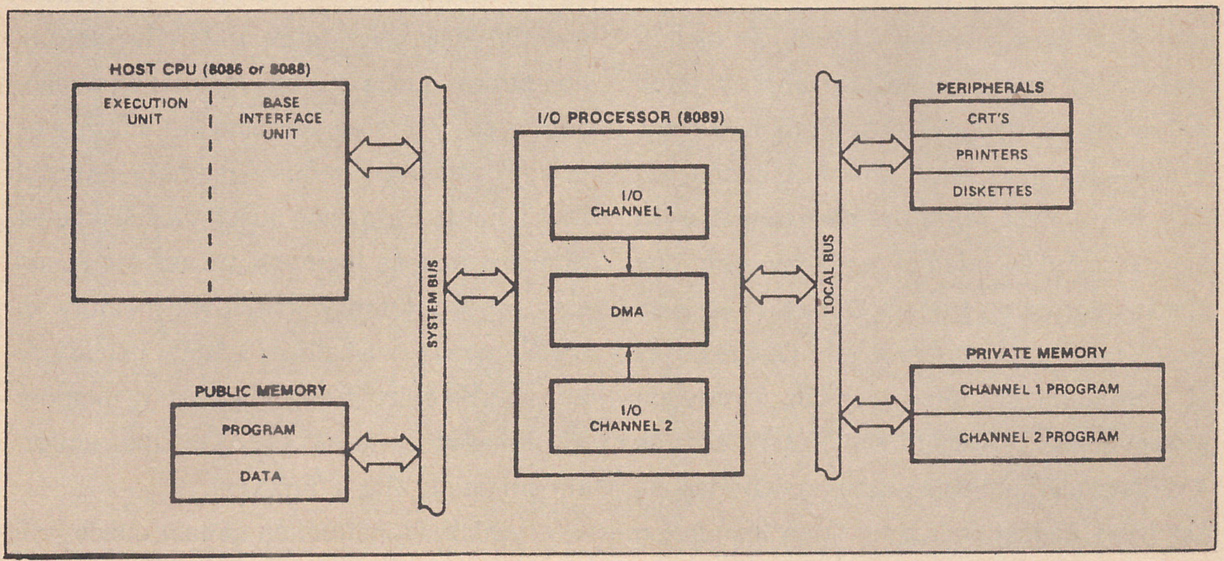
8086,8088 เป็นไมโครโปรเซสเซอร์ ที่มีขีดความสามารถสูงมากตัวหนึ่งซึ่งจากการวัด performance หรือประสิทธิภาพในการทำงานอาจจะบอกได้ว่า 8086, 8088 มี performance ดีกว่า 8080 ซึ่งเป็น CPU แบบ 8 บิท ประมาณ 6-10 เท่าจุดหนึ่งที่ทำให้ 8086, 8088 มี performance สูงก็คือลักษณะสถาปัตยกรรมภายในของตัว CPU ซึ่งมี 16 bit internal data path และลักษณะของ pipelined architecture ภายในซึ่งทำให้ CPU สามารถ fetch instruction จาก memory ได้ในขณะที่ CPU กำลัง execute อยู่ใน machine cycle ที่ไม่ต้องการ access memory และการวางลักษณะ format ของ opcode ที่เหมาะสม ทำให้ CPU สามารถ fetch instruction ได้มากขึ้นในช่วงเวลาเดียวกันเมื่อเทียบกับไมโครโปรเซสเซอร์เบอร์อื่น ๆ

ซอฟท์แวร์ซึ่งใช้กับ 8086,8088 นั้นไม่จำเป็นต้องเขียนด้วยภาษา Assembly เสมอไป CPU 8086,8088 นั้นออกแบบมาเพื่อให้มีลักษณะทางฮาร์ดแวร์ที่จะสนับสนุนโปรแกรมที่เขียนด้วย high level language เช่น PL/M-86 ของ INTEL เอง 8086,8088 นั้นมีชุดคำสั่งซึ่งสามารถทำ operation ต่าง ๆ กับ operand ที่อยู่บน memory ได้โดยตรง วิธีการอ้างแอดเดรส (addressing mode) ของ operand มีถึง 24 mode ซึ่งสามารถอ้างถึง effectived address คือแอดเดรสที่จะส่งไปยัง memory จริง ได้โดยใช้ค่า Base Register,Index Register และ Displacement ซึ่งเก็บไว้ใน Instruction เองมาประกอบกันเป็น effective address ได้ ลักษณะของ addressing mode ดังกล่าวคล้ายกับลักษณะของภาษา Assembly Language ของคอมพิวเตอร์ขนาดใหญ่ เช่น IBM เป็นต้น

โครงสร้างสถาปัตยกรรมภายในของ 8086,8088

ดังที่บอกมาแล้วว่าลักษณะโครงสร้างภายในของ 8086,8088 ที่ดีมีส่วนเพิ่ม Performance ของ CPU ได้มาก ต่อไปเราจะมาศึกษาถึงลักษณะพิเศษของสถาปัตยกรรมภายใน ซึ่งทำให้ CPU มี Performance สูง

ลักษณะการ Execute คำสั่งของไมโครโปรเซสเซอร์ทั่ว ๆ ไปนั้นสามารถแสดงได้ดังรูปที่ 3 ซึ่งเราจะเห็นได้ว่าไมโครโปรเซสเซอร์จะต้อง fetch เอาคำสั่งที่จะ execute มาจาก memory และหลังจากนั้นก็จะ decode คำสั่งแล้ว execute ตามคำสั่งที่ fetch มา หลังจากที่ execute คำสั่งเสร็จ จึง fetch คำสั่งต่อไปมา



รูปที่ 3 โครงสร้างภายในของ 8089 I/O Processor

execute ต่อไป และจะเป็นอย่างนี้เรื่อย ๆ ซึ่งเราจะเห็นว่าส่วน hardware ของ CPU ที่ทำหน้าที่ execute คำสั่งต่าง ๆ จะต้องรอจนกว่า intruction จะถูก fetch เข้ามาใน CPU และทำการ decode ก่อนจึงจะเกิดการ execute ขึ้น ดังนั้นส่วน EXECUTION HARDWARE คือ ALU และส่วนวงจรควบคุม (CONTROL CIRCUITARY) จะต้องเสียเวลาคอยจนกว่าจะเกิดการ execute เสมอ แต่ 8086,8088 ได้ออกแบบโครงสร้างภายใน CPU ทำให้สามารถลดหรือขจัดเวลาที่ต้องรอคอยของส่วน EXECUTION HARDWARE นี้ลง โดยโครงสร้างภายในของ 8086, 8088 สามารถแบ่งออกเป็น 2 Processing Unit ซึ่งแยกออกจากกัน ทั้งสอง Processing Unit สามารถทำงานได้พร้อม ๆ กัน และเป็นอิสระต่อกัน ทั้งสองหน่วยนี้คือ

- **EXECUTION UNIT (EU)** *ซึ่งทำหน้าที่* ***Execute*** *คำสั่งที่ถูก* ***fetch*** *เข้ามา*
- **BUS INTERFACE UNIT (BIU)** *ทำหน้าที่* ***fetch*** *เอา* ***opcode*** *ซึ่งอยู่ใน* ***memory*** *เข้าสู่* ***CPU*** *โดยทางผ่าน* ***data bus*** *และทำหน้าที่* ***transfer*** *ข้อมูลระหว่าง* ***Execution Unit*** *กับ* ***Memory*** *และ* ***I/O device***

จากการที่โครงสร้างภายในของ 8086,8088 ถูกแบ่งออกเป็น EU และ BIU ซึ่งสามารถทำงานเป็นอิสระต่อกัน ทำให้ 8086,8088 สามารถทำงานในช่วงเวลาที่ซ้อนทับ (Overlap) กัน ระหว่างการ Execution และ การ fetch คำสั่งได้ คือขณะที่ส่วน EU ทำการ execute คำสั่งปัจจุบันอยู่ส่วน BIU จะทำการ fetch เอา opcode ของคำสั่งต่อไปจาก memory มาเก็บเข้าไว้ใน CPU ในเวลาเดียวกัน ซึ่งจะทำให้เวลาที่ใช้ในการรอคอยการ fetch คำสั่งต่อไปของ EU ถูกขจัดออกไป EU จะสามารถ execute คำสั่งต่อไปซึ่งถูก BIU fetch เข้ามาเก็บไว้ใน CPU แล้วได้ทันที คำสั่งที่จะถูก fetch ผ่าน data bus เข้ามาโดย BIU และถูกเก็บไว้ใน internal ram array ภายใน CPU เรียกว่า Instruction Stream Queue ซึ่งมีลักษณะโครงสร้างของ data structure แบบ Queue EU จะใช้ instruction ที่ถูก fetch มาเก็บไว้ใน instruction stream queue มา execute ต่อทันทีหลังจาก execute คำสั่งปัจจุบันเสร็จแล้ว ซึ่งลักษณะของการ Overlap กันระหว่างการ Execution และการ fetch instruction สามารถแสดงได้ดังรูปที่ 5 และรูปที่ 6 แสดงการเปรียบเทียบการ execute คำสั่ง 3 คำสั่งระหว่าง 8086,8088 กับ ไมโครโปรเซสเซอร์อื่น ๆ ซึ่งแสดงให้เห็นถึง performance ที่ดีกว่าของ 8086,8088

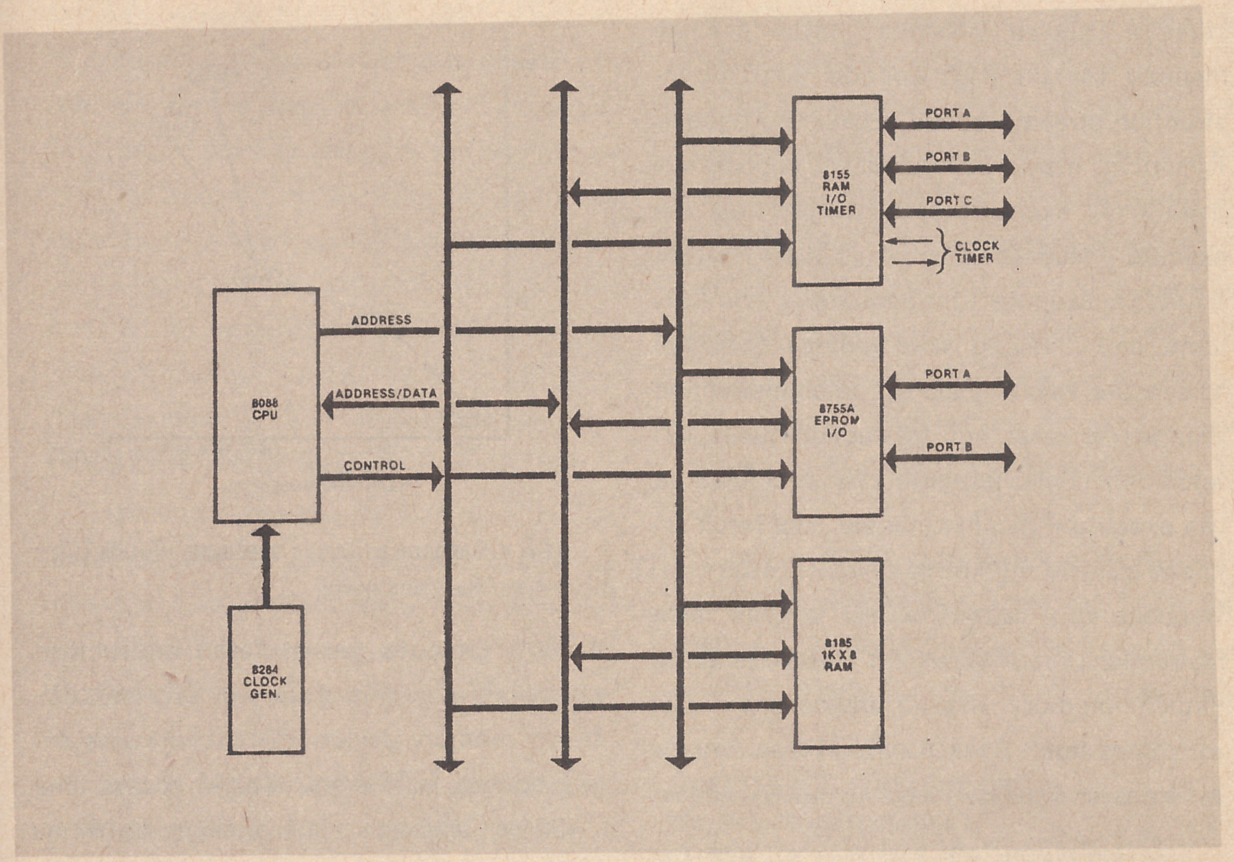
ลักษณภายในของ 8086,8088

- ลักษณะโครงสร้างของ EU นั้นประกอบด้วย 16 bit arithmetic/Logic Unit (ALU) ซึ่งทำหน้าที่ควบคุม status ของ CPU และ flag ต่าง ๆ ทำหน้าที่ควบคุมค่าใน general register ต่าง ๆ ตามคำสั่งที่ execute ส่วนทางเดินของข้อมูล (data path) และ register ภายใน EU มีขนาด 16 bit

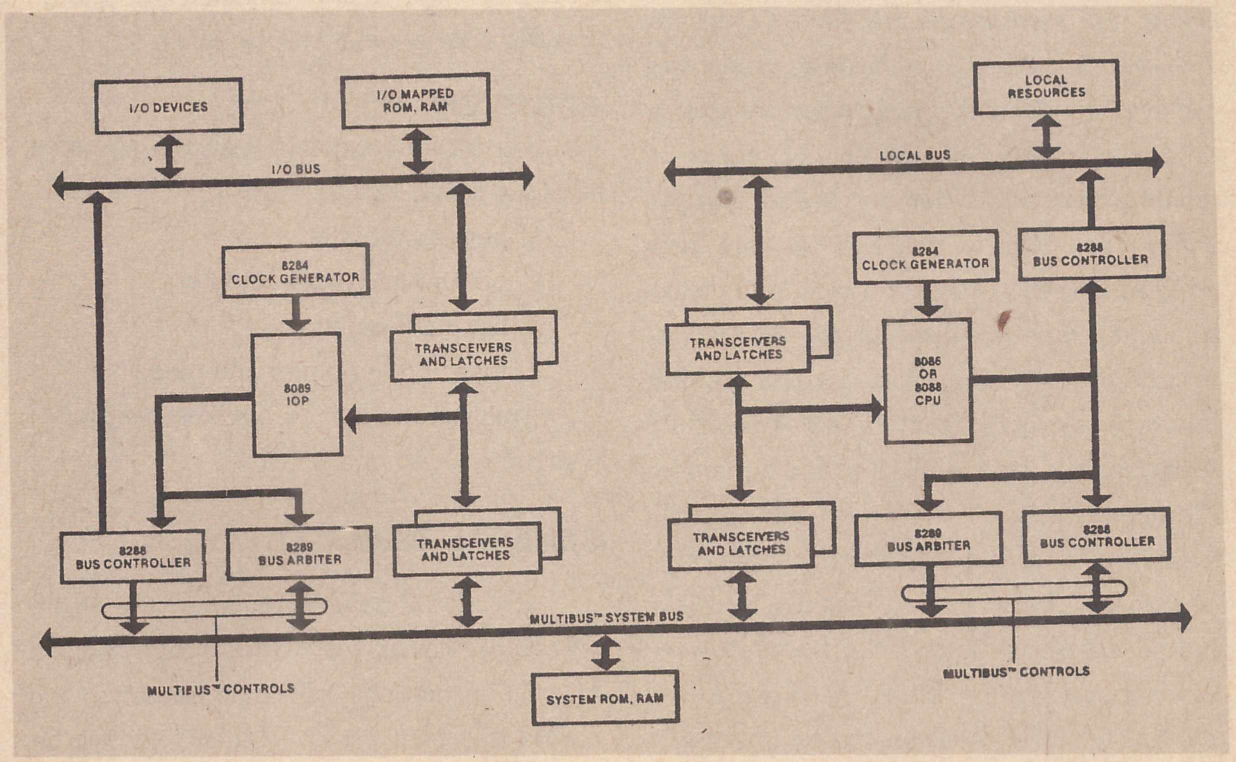
EU ไม่มีส่วนติดต่อกับ System Bus โดยตรง แต่รับคำสั่งที่จะทำจาก instruction stream queue ซึ่ง BIU จะทำหน้าที่ fetch เอาคำสั่งมาเก็บไว้ใน queue แต่ถ้าคำสั่งที่ execute มีการ access ค่าใน memory หรือ I/O device EU จะ transfer ค่ากับ memory หรือ I/O โดยผ่านทาง BIU จะทำการ Read หรือ Write ตามแอดเดรสที่ EU ต้องการลักษณะ EU ของ 8086, 8088 เหมือนกันทุกประการ ถึงแม้ว่า external data path จะมีขนาดต่างกัน

- ลักษณะโครงสร้างของ BIU ของ 8086, 8088 จะมี function การทำงานเหมือนกัน แต่ถูก implement ต่างกัน เพื่อให้สอดคล้องกับลักษณะของ data bus จากที่กล่าวมา BIU จะทำหน้าที่ Bus operation แก่ EU ทำหน้าที่เคลื่อนย้ายข้อมูลระหว่าง EU กับ memory และ I/O ตาม address ที่ EU ต้องการ BIU จะ fetch คำสั่งต่อ ๆ ไปจาก memory ในขณะที่ EU กำลัง execute คำสั่งปัจจุบันอยู่ โดยจะ fetch มาเก็บไว้ใน instruction stream queue ขนาดของ instruction stream queue ของ 8088 มีขนาดยาวเพียง 4 ไบท์ 8088 จะทำการ fetch คำสั่งต่อไปจาก memory ทันที ถ้ามีที่ว่าง 1 ไบท์ ใน instruction queue และไม่มี request เพื่อทำการ access memory และ I/O จาก EU แต่ 8086 จะทำการ fetch คำสั่งต่อไปจาก memory มาเก็บไว้ใน queue ก็ต่อเมื่อมีที่ว่าง 2 ไบท์ใน queue และในขณะนั้นไม่ต้องการที่จะเคลื่อนย้ายข้อมูลกับ memory หรือ I/O 8086 จะ fetch คำสั่งเข้ามา instruction stream queue ทีละ 2 ไบท์เสมอ

โดยปกติแล้ว instruction stream queue จะมี opcode ของคำสั่งเก็บไว้ภายใน queue อย่างน้อย 1



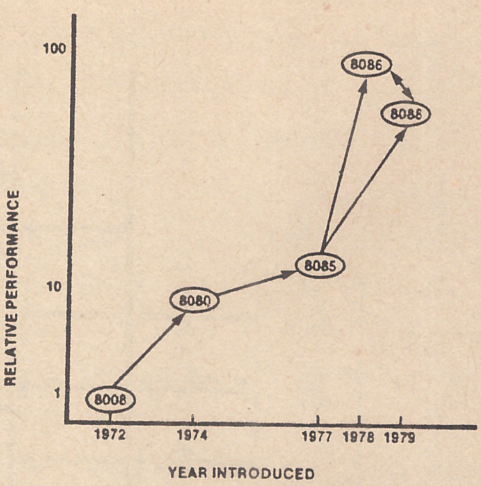
รูปที่ 4 ระบบไมโครคอมพิวเตอร์ขนาดเล็กที่ใช้ 8088



รูปที่ 5 ระบบ Multiprocessing ที่ประกอบด้วย 8086/8088/8089

ไบท์เสมอ ดังนั้น EU ไม่ต้องรอการ fetch คำสั่งจาก memory ของ BIU คำสั่ง opcode ที่เก็บไว้ใน instruction stream queue จะเหมือนกับ opcode ที่ถูกเก็บไว้ใน memory ที่ถูกเก็บต่อจาก opcode ของคำสั่งที่กำลัง execute อยู่ขณะนั้น แต่ถ้า EU เกิด execute คำสั่งซึ่งต้อง ย้ายการทำงานไปยังตำแหน่งอื่น ซึ่งไม่ใช่การ execute แบบ sequential เช่น คำสั่ง jump, call เป็นต้น EU จะส่ง address ของคำสั่งที่จะ execute ต่อไปให้แก่ BIU เพื่อ fetch instruction เข้ามา BIU จะ reset ค่าใน instruction queue แล้ว fetch เอาคำสั่งที่จะ execute เข้ามาแล้วส่งให้แก่ EU เพื่อ execute ทันที หลังจากนั้น BIU ก็เริ่ม fetch เอาคำสั่งซึ่งเป็นคำสั่งที่ถูกเก็บต่อจากแอดเดรสนั้นเข้ามาไว้ใน queue ต่อไป ในอีกกรณีหนึ่ง BIU จะหยุดการ fetch instruction ถ้า EU ต้องการทำการ Read หรือ Write ข้อมูลกับ memory หรือ I/O device BIU ก็จะทำกระบวนการ fetch ที่กำลังทำอยู่ให้เสร็จก่อน จึงจะเริ่มการ transfer ข้อมูลระหว่าง EU กับ memory หรือ I/O

อีกจุดหนึ่งที่น่าสนใจคือ 8086,8088 สามารถทำงานด้วยสัญญาณนาฬิกาสูงสุดถึง ***5 MHz*** *แต่* ***interface*** *กับ* ***memory device*** *ที่มีความเร็วในการทำงานต่ำๆ เช่น* ***RAM, EPROM*** *ซึ่งมี* ***memory time ≥ 450 ns*** *ได้ทันทีโดยไม่ต้องต่อวงจรซึ่งทำหน้าที่เพิ่มเติม* ***wait state*** *ใน* ***machine*** cycle ของการ Read หรือ Write (memory access time หมายถึงเวลาที่ใช้หลังจากที่ CPU ให้แอดเดรส แก่ memory device แล้ว memory device ให้ข้อมูลที่ถูกต้อง (Valid data) แก่ data bus ของ CPU) ในการทำ Read Operation Memory device ที่ยิ่งมี memory access time ต่ำ (มีความเร็วสูง) จะยิ่งมีราคาสูง จากรูปที่ 7 แสดงการเปรียบเทียบ 8086,8088 กับ 8 bit microprocessor อื่น ๆ เช่น z-80B ซึ่งสามารถใช้สัญญาณนาฬิกา สูงสุด 6 MHz แต่ต้องใช้ memory ที่มี memory access time น้อยกว่า 140 ns เท่านั้น จึงต้องเพิ่มส่วนวงจรเพิ่ม wait state แก่ CPU ซึ่งทำให้เสียค่าใช้จ่ายเพิ่มขึ้น จะเห็นว่า 8086,8088 สามารถให้ full performance ในขณะที่ใช้ memory device ที่มีความเร็วต่ำและถูกกว่า ไมโครโปรเซสเซอร์อื่น ๆ ที่สามารถทำเช่นนั้นได้ มีผลมาจากการออกแบบโครงสร้างภายในที่แยกกันเป็น EU และ BIU ของ 8086,8088 ซึ่ง



รูปที่ 6 ขีดความสามารถของ 8086/8088 เทียบกับไมโครโปรเซสเซอร์อื่น ๆ ของ INTEL

ทำให้การ Execute และการ fetch instruction สามารถทำได้พร้อม ๆ กัน ดังนั้น BIU จึงไม่จำเป็นที่จะต้องการ memory device ที่มีความเร็วสูง 8088 สามารถ operate โดยใช้ EPROM ซึ่งปกติมี access time 450 ns ได้โดยตรง ในที่ภายใต้สัญญาณนาฬิกาสูงสุดส่วน 6809 หรือ z-80B จะต้องเพิ่ม Wait state ใน machine cycle เมื่อต้องต่อกับ memory device ประเภท EPROM ขณะที่ทำงานด้วยสัญญาณนาฬิกาสูงสุดคือ 4 MHz และ 6 MHz ตามลำดับ

REGISTER

เราสามารถแบ่ง register ภายในของ 8086,8088 ออกเป็น 4 group คือ

1. general register
2. pointer and index register
3. segment register
4. instruction pointer and feags register

ต่าง ๆ ภายใน 8086,8088 สามารถแสดงได้ดังรูปที่ 7, 8

GENERAL REGISTER

8086,8088 มี general register ขนาด 16 bit 4 ตัว คือ AX,BX,CX,DX, โดยทั่วไปสามารถใช้เป็น operand ในการทำ arithmatics หรือ logic operation เช่น คำสั่งจำพวกบวก-ลบ คูณ-หาร หรือการทำ NOT AND OR ระหว่าง register register 16 bit แต่ละตัวสามารถ

มองได้เป็น register ขนาด 8 bit 2 ตัว มาประกอบกัน เราสามารถทำ 8 bit operation บน high byte หรือ low byte ของ register ขนาด 16 bit ตัวใดตัวหนึ่ง ได้โดย high byte และ low byte ของ register แต่ละตัวมีชื่อเรียกเฉพาะ เช่น *AL หมายถึง **low byte** ของ **register** AX และ AH หมายถึง **high byte** ของ AX* เป็นต้น register บางตัว จะมีหน้าที่เฉพาะในคำสั่งบางคำสั่ง เช่น CX จะถูกใช้เป็น loop counter ในคำสั่ง LOOP หรือ Bx ในคำสั่ง XLAT ซึ่ง Bx ถูกใช้ชี้ไปยังจุดเริ่มต้นของ table ที่ใช้ในการ translate เป็นต้น

POINTER AND INDEX REGISTER

ประกอบด้วย register ขนาด 16 bit 4 ตัว : **SP (stock pointer), BP (base pointer), SI (Source index), DI (destination index)** register เหล่านี้สามารถนำมาใช้เป็น operand ของ arithmatic หรือ logic operation ได้เช่นกัน แต่ทำ operation ได้ทีละ 16 bit เท่านั้น

*SP เป็น **register** ทำหน้าที่เป็น **pointer** ชี้ไปยัง **top of stack** ใน **stack segment** ซึ่งจะทำหน้าที่คล้ายกับ* SP ของ 8-bit microprocessor ทั่วไปและจะถูกเปลี่ยนค่าโดยคำสั่งเกี่ยวกับ stack operation เช่น คำสั่ง PUSH,POP หรือ CALL เป็นต้น

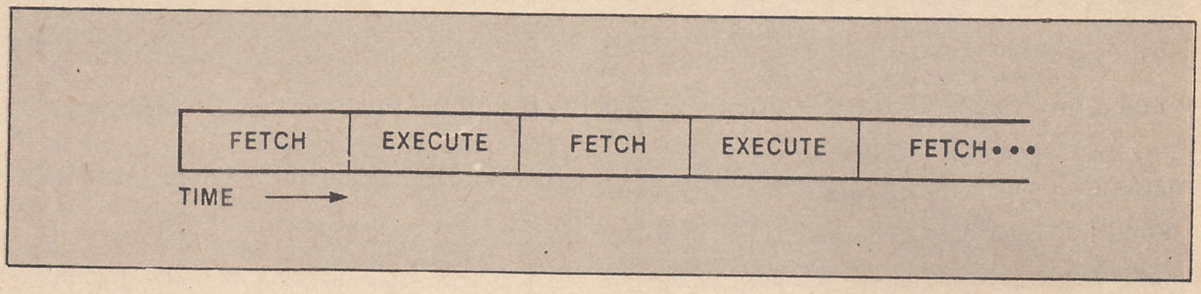
*BP เป็น **register** ที่ทำหน้าที่เป็น **pointer** ที่จะชี้ไปยังจุดใดจุดหนึ่งใน **stack segment** ตามที่เราต้องการ* ซึ่งมีประโยชน์มาก เราอาจใช้ BP เป็นตัวชี้ไปยังจุดเริ่มต้นของ parameter list บน stack segment ขณะที่เราทำ procedure call ซึ่งมีการส่งผ่าน parameter ของ subroutine ผ่านทาง stack BP อาจจะใช้เก็บค่าของ Stack pointer เดิมก็ได้และยังสามารถใช้เป็น base register ใน addressing mode แบบ base addressing หรือ base indexed addressing อีกด้วย

*SI และ DI เป็น **register** ที่ทำหน้าที่เป็น **index register** ใน **addressing mode** แบบ **index addressing** หรือ **base indexed addressing** และจะถูกใช้เป็น **index** ในคำสั่งประเภท **string operation***

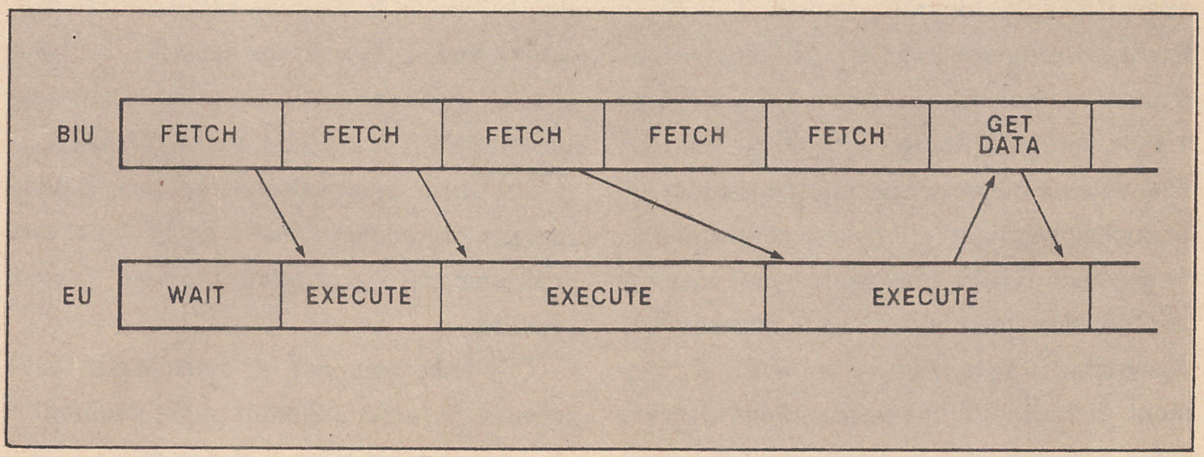
SEGMENT REGISTER

Memory Segmentation

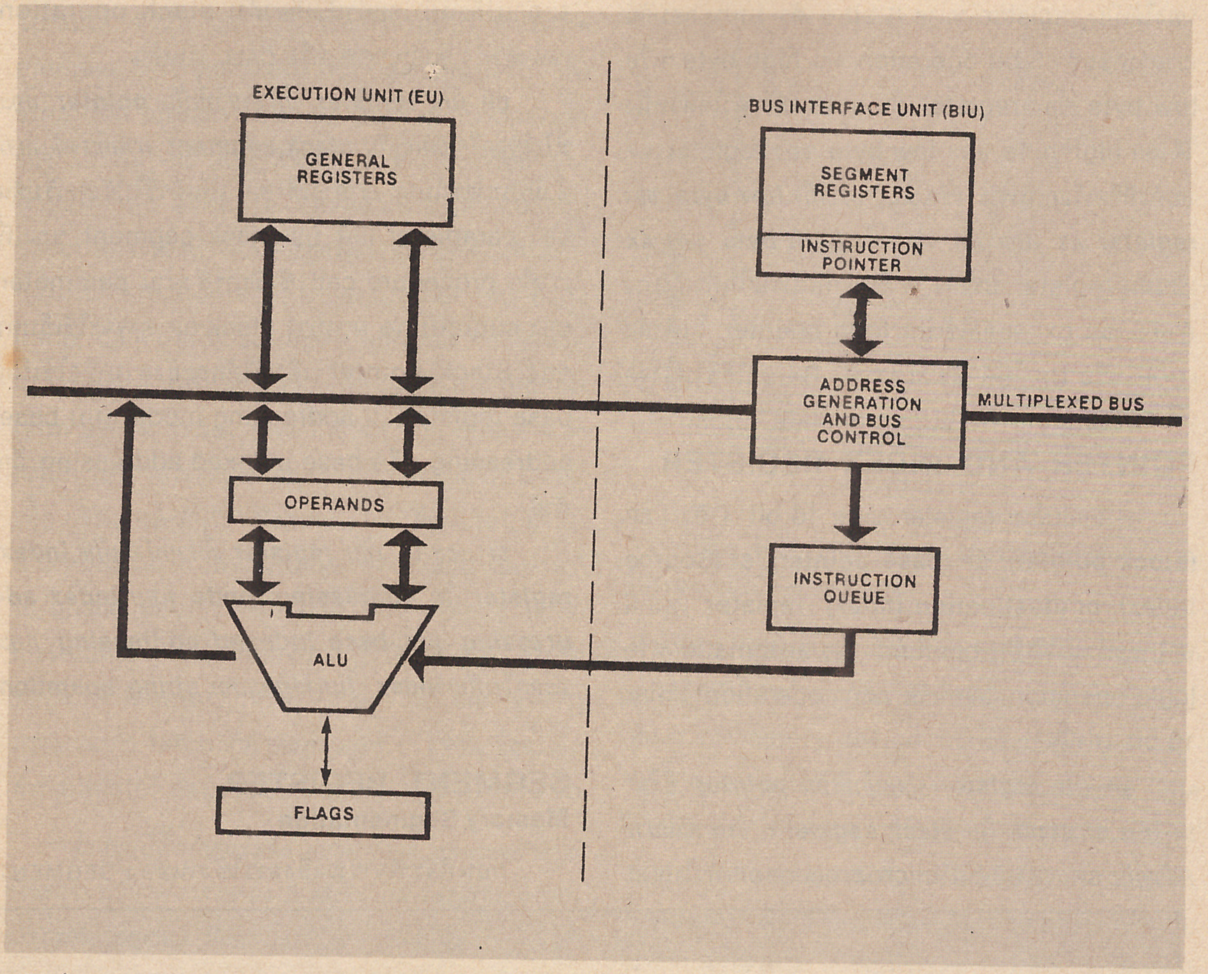
ก่อนที่จะพิจารณาถึงการใช้งานของ SEGMENT



รูปที่ 7 ลำดับการทำงานในไมโครโปรเซสเซอร์ทั่วไป



รูปที่ 8 การทำงานแบบขนาน (พร้อมกัน) ของ BIU และ EU ใน 8086/8088



รูปที่ ๑ โครงร่างภายในของ 8086/8088 ประกอบด้วย Execution Unit และ Bus Interface Unit (EU และ BIU)

REGISTER เราจะศึกษาถึงลักษณะ memory segmentation ของ 8086,8088 เสียก่อน ดังที่ได้กล่าวมาแล้วว่า 8086,8088 สามารถอ้างถึง memory space ได้ทั้งหมด 1 Megabyte บน memory space ทั้งหมดจะถูกแบ่งออกเป็น logical segment ต่าง ๆ ซึ่งถูกกำหนดโดย program แต่ละ logical segment คือส่วนของ memory space ที่ต่อกันซึ่งมีขนาดได้ตั้งแต่ 1 byte จนถึง 64 Kbyte และสามารถกำหนดให้อยู่ตำแหน่งใดบน memory space ก็ได้ตามต้องการ แต่ละ logical segment จะมี base address เพื่อบอกว่า segment นี้เริ่มต้น ณ address ใดในตัว memory จริง (physical memory) ซึ่ง base address นี้จะต้องตกอยู่บน 16 byte memory boundary คือ segment จะต้องเริ่มต้นที่ physical address ซึ่งถูกหารด้วย16 ลงตัวเสมอ แต่ละ segment ที่เรากำหนดขึ้นนั้นสามารถอยู่ติดกัน (contiguous) แยกกัน (disjoint) หรือ ซ้อนกัน (overlap) กับ segment อื่นได้ดังแสดงในรูป

ในขณะใดขณะหนึ่ง CPU จะสามารถ access ค่าที่เก็บอยู่ใน segment ได้ทั้งหมด 4 segment ใน memory ซึ่งมีชื่อเรียกคือ **CODE SEGMENT,DATA SEGMENT, EXTRA SEGMENT,STACK SEGMENT** โดย base address ของ segment เหล่านี้จะถูกเก็บอยู่ใน segment register ขนาด 16 bit 4 ตัว ดังรูป คือ...

CS (code segment register)** จะเก็บค่า **base address** ของ **code segment** ซึ่งเป็นที่เก็บของ **program op.code** ซึ่งจะถูก **fetch** ไปโดย **BIU** เพื่อทำการ **execute

DS (data segment register)** จะเก็บค่า **base address** ของ **data segment** ซึ่งเป็น **segment** ที่เก็บค่าของ **variable** ต่าง ๆ ใน **program

SS (stack segment register)** จะเก็บค่า **base

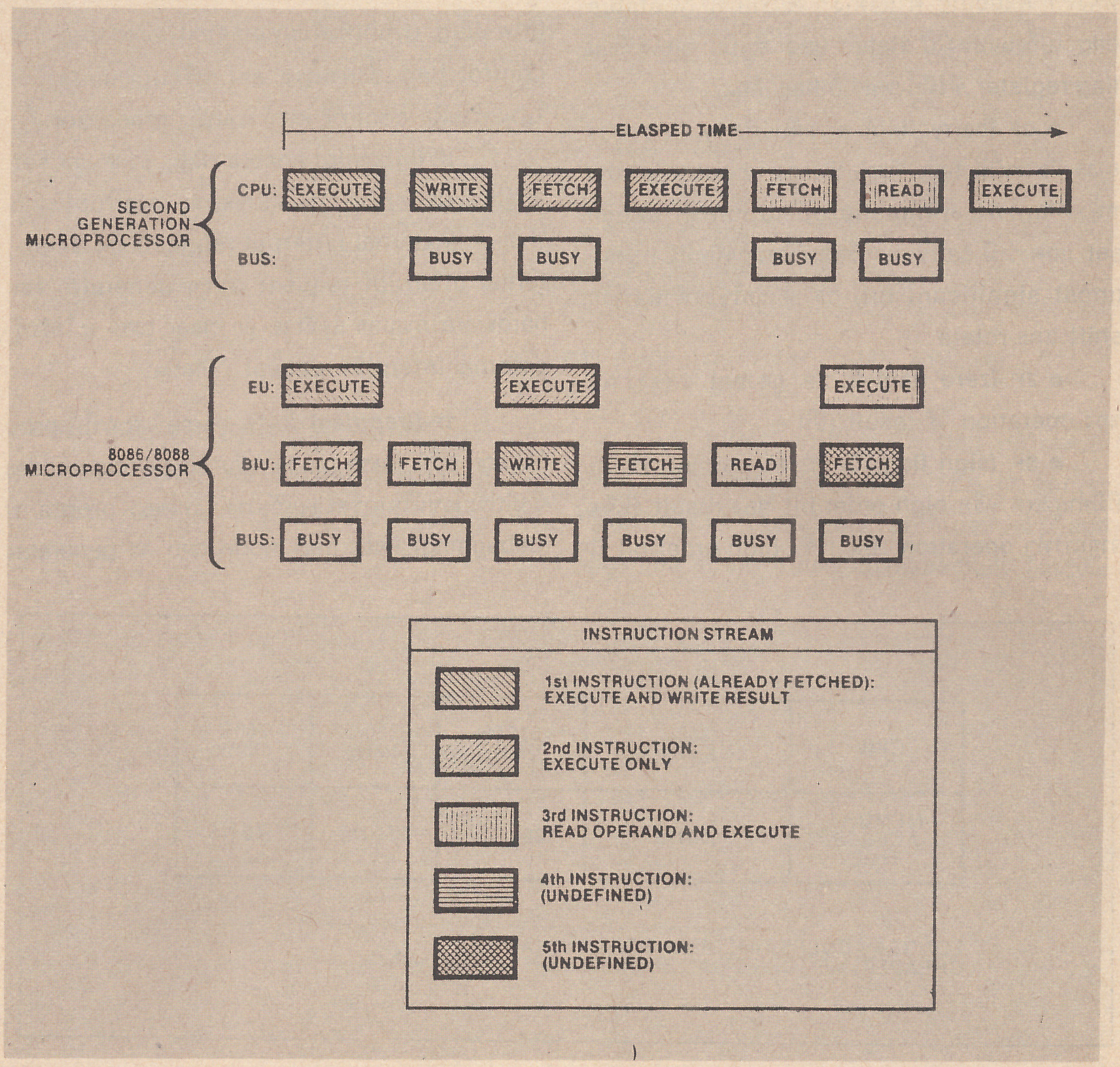
address *ของ* ***stack segment*** *ซึ่งเป็น* ***segment*** *ที่เก็บค่าอันเนื่องมาจากการทำ* ***stack operation*** *เช่นค่าที่เก็บใน* ***register*** *ในคำสั่ง* ***PUSH*** *หรือเก็บค่าของ* ***return address*** *ในการทำคำสั่ง* ***CALL*** *เป็นต้น โดยใช้ร่วมกับ* ***SP*** *และ* ***BP*** *ในการทำ* ***stack operation*** *ดังกล่าว*

ES (extra segment register) *จะเก็บค่า* ***base address*** *ของ* ***extra segment*** *ซึ่งเป็น* ***segment*** *ที่คล้ายกับ* ***data segment*** *คือสามารถใช้เก็บค่า* ***variable***

ค่าที่เก็บใน segment Regsiter ทั้ง 4 สามารถถูกเปลี่ยนแปลงค่าโดยใช้คำสั่งในโปรแกรม ทำให้เราสามารถ set ให้ segment ทั้ง 4อยู่ที่ใดบนmemory space ได้ตามต้องการ ลักษณะ segment structure ของ 8086/8088 จะทำให้การออกแบบระบบจะเป็นลักษณะ modular software design และสามารถใช้งานได้อย่างสะดวกในหลาย application เช่น ในกรณีของโปรแกรมแก้ไขข้อมูล (editor program) สำหรับหลาย ๆ on-line terminal เราสามารถ run โดยใช้โปรแกรมบน code segment เดียวกันแล้วจัดให้มี buffer สำหรับแต่ละ terminal ทำให้ program สามารถทำงานให้ user แต่ละคนได้ง่ายโดยเพียงแต่เปลี่ยนค่า ESให้ point ไปยัง buffer ของแต่ละ terminal ตามต้องการแล้ว process แก้ไขข้อมูลบน buffer นั้น

INSTRUCTION POINTER AND FLAG

IP (instruction pointer) *เป็น* ***register*** *ขนาด*



รูปที่ 10 การเหลื่อมซ้อนกัน (overlapped) ของการทำงานในช่วง fetch และ execute

16 bit *ซึ่งทำหน้าที่คล้ายกับ* ***program counter*** *ใน* ***8-bit microprocessor*** *ทั่วไป* IP จะถูก update ค่าโดย BIU ใน IP จะเก็บค่า offset ของคำสั่งต่อไปที่จะถูก fetch โดยคิดจาก base address ของ code segment ค่าของ IP อาจจะถูก PUSH ลง stack หรือ restore จาก stack ได้ IP จะถูกเปลี่ยนแปลงโดยคำสั่งพวก CALL หรือ JUMPหรือเกิดจาก interrupt ขึ้น BIU จะใช้ค่าของ CS และIP เพื่อประกอบกันเป็น physical address ของคำสั่งที่ต้องการ fetch *(ซึ่งจะถูกกล่าวถึงในหัวข้อ* ***physical address generation*** *ต่อไป)*

flag register 8086-8088 มี ***status flag*** *ที่แสดงถึงสถานะหรือผลการทำ* ***operation*** *ต่าง ๆ ของ* ***CPU*** หลังจาก execute คำสั่งต่าง ๆ จะมีผล ทำให้ status flag เปลี่ยนแปลงไป status flag จะเป็น bit หนึ่งใน flag register ซึ่งมีทั้งหมด 6 flag คือ...

● **CF (carry flag)** คล้ายกับ carry flag ของ 8080 คือจะเป็น flag ที่แสดงถึง carry out หรือตัวทดหลังจากการทำ arithmatic operation ถ้า CF ถูก set แสดงว่ามี carry ทดหรือถูกยืมมาจากบิทที่มีค่าสูงสุด (most significant bit) CF ยังถูกใช้ในคำสั่งพวก shift และ rotate

● **ZF (zero flag)** ถ้า ZF ถูก set แสดงว่าผลของ operation ได้ result เป็น 0

● **SF (sign flag)** ถ้า SF ถูก set แสดงว่าบิทที่มีค่าสูงสุด หรือ high order bit ของ result ที่เกิดจากการทำ operation เป็น 1 ซึ่ง high order bit นี้จะใช้เป็น Sign bit ในระบบเลข two complement

● **OF (overflow flag)** ถ้า OF ถูก set แสดงว่าเกิด overflow ขึ้นเนื่องจากการทำ arithmatic operation

● **AF (auxiliary flag)** ถ้า AF ถูก set แสดงว่ามีการยืมหรือทดไประหว่าง bit 3 bit 4 ของ operand ในการทำ 8-bit arithmatic operation AF มักจะถูกใช้ทำในคำสั่งพวก decimal arithmatic instruction

● **PF (parity bit)** ถ้า PF ถูก set แสดงว่า result ของ operation นั้นมีจำนวน bit ที่เป็น 1 เป็นเลขคู่ (even parity)

เราจะเห็นว่า status flag เหล่านี้คล้ายกับ flag ต่าง ๆ ของ 8080 หรือ Z80 มาก มีเพียง OF หรือ overflow flag เท่านั้นที่เพิ่มมานอกจากนี้ 8086-8088 ยังมี control flag ซึ่งสามารถ set หรือเปลี่ยนแปลงโดยโปรแกรมเพื่อ control การทำงานของ processor คือ

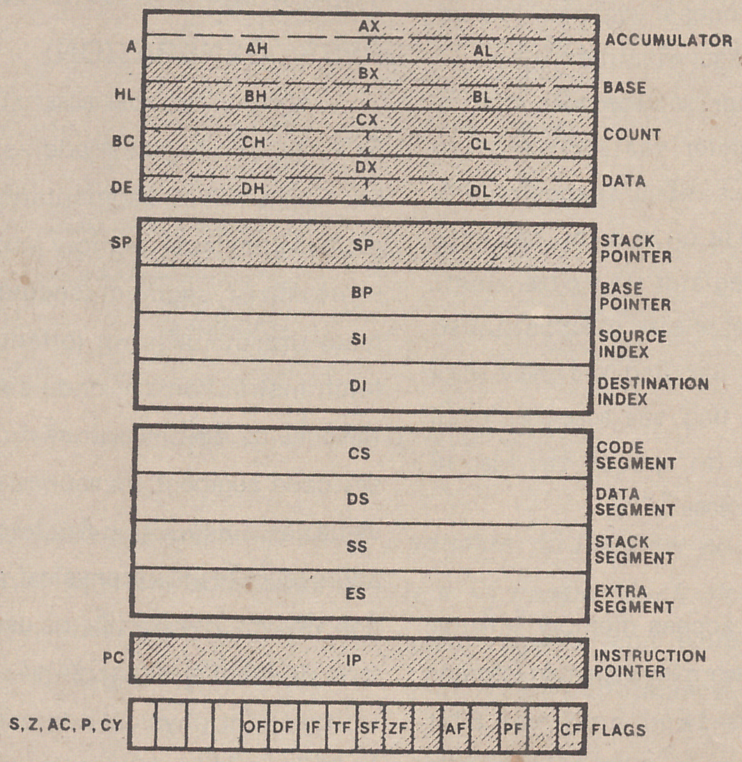
☐ **IF (interrupt-enable flag)** ถ้า IF ถูก set จะทำให้ CPU จะรับรู้การเกิดสัญญาณ interrupt จากภายนอก (external interrput request) ขึ้น ซึ่ง external interrupt request จะถูก generate จาก hardware ภายนอก แต่ถ้า IF ถูก clear CPU จะไม่รับรู้ external interrupt request ที่เกิดขึ้น

☐ ***TF (trap flag)*** ถ้า TF ถูก set จะทำให้ processor อยู่ใน trace mode หรือ single-step mode ซึ่งมีประโยชน์ในการตรวจแก้การทำงานของ program (debugging) คือใน trap mode CPU จะ generate

CPU	8088 5MHz	68B09 2MHz	Z80A 4MHz	Z80B 6MHz
MEMORY ACCESS TIME	460 NS	320 NS	250 NS	140 NS

LONGER ACCESS TIME MEANS SLOWER (AND CHEAPER) MEMORIES CAN BE USED WITH iAPX 88

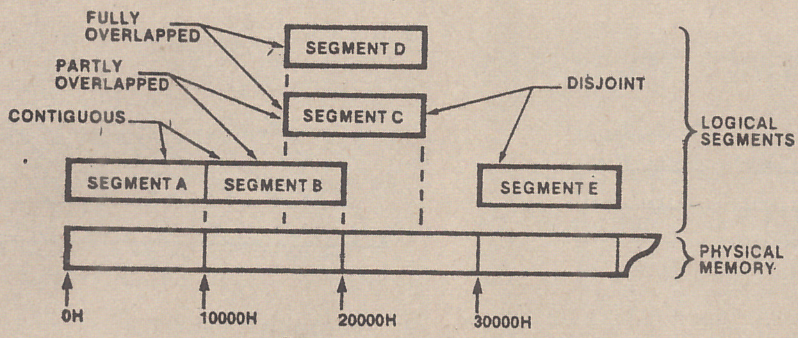
รูปที่ 11 ช่วงเวลาในการ Access Memory ของไมโครโปรเซสเซอร์เบอร์ต่างๆ



รูปที่ 12 โครงสร้าง Register ภายใน 8086/8088 ส่วนที่แรเงาหมายถึงซ้ำกับ ที่มีใน 8080/8085

internal interrupt อย่างอัตโนมัติหลังจากที่ execute แต่ละคำสั่งเสร็จซึ่งทำให้เราสามารถตรวจสอบการทำงานหรือค่าของ register ต่าง ๆ ขณะที่ทำงานแต่ละคำสั่งจบลง

☐ ***DF (direction flag)*** เป็น flag ที่กำหนดทิศทางในการทำ string operation ถ้า DF ถูก set จะทำให้คำสั่งที่ทำ string operation จะประมวลผลข้อมูลที่เป็น string แบบ auto-decrement คือจะเริ่ม



รูปที่ 13 ตำแหน่งของ segment ใน physical memory

process จาก high address ไปสู่ low address หรือแบบขวาไปซ้าย แต่ถ้า DF ถูก clear จะทำงานในลักษณะตรงข้ามคือจะ process โดยเริ่มจาก low address ไปสู่ high address หรือแบบซ้ายไปขวา

จากลักษณะ register ของ 8086-8088 เราจะเห็นว่ามีส่วนสัมพันธ์ กับ register ของ 8080-8085 เราอาจจะมอง AL ของ 8086-8088 เหมือนกับ accumulator ของ 8080-8085 และ BH,BL,CH,CL,DH,DL ของ 8086-8088 สามารถมองเป็น register H,L,B,C,D,E, ได้ตามลำดับ IP และ SP ของ 8086-8085 อาจจะมองเป็น PC และ SP ของ 8080-8085 และ flag ต่าง ๆ ของ 8080-8085 เหมือนกับ status flag ของ 8086-8088 ดังนั้นเราสามารถจะ translate เอา opcode ของ 8080 ให้เป็น opcode ของ 8086-8088 ได้ง่าย

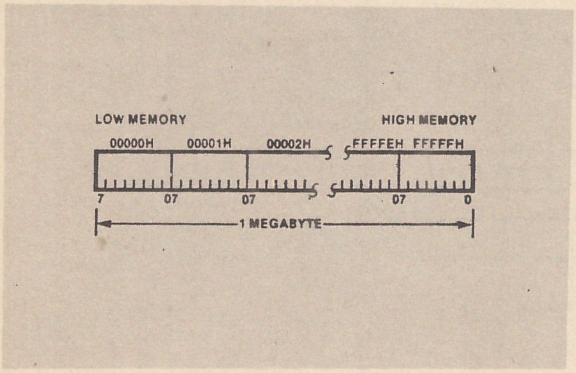
MEMORY

8086-8088 สามารถ access memory ได้ทั้งหมด 1 Megabyte โดยมีลักษณะการจัดหน่วยความจำ (storage organization) ที่จะมองคล้ายกับ array ของ 8-bit byte ที่เรียงต่อกัน คำสั่ง (op-code), byte data และ word data สามารถเก็บไว้ ณ address ใดใน memory ก็ได้จึงเป็น Unaligned word variable คือไม่ต้องตกอยู่บน word boundary (หารด้วย 2 ลงตัว) เพราะ 2 byte เป็น 1word เหมือนกับ 16 bit microprocessor บางตัว เช่น MC 68000เป็นต้น ทำให้ code ของ program สามารถ pack เก็บต่อกันตลอดไม่เปลือง memory มากนัก word variable ก็ไม่ต้องตกบน word boundary และลักษณะการเก็บจะเหมือนกับของ 8080 คือจะเก็บ least singificant byte มาก่อนแล้วตามด้วย most significant byte

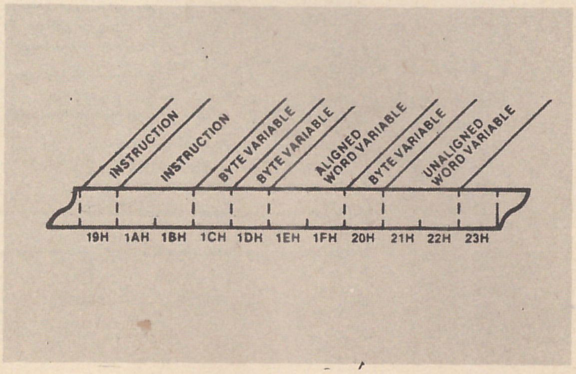
การกำหนดแอดเดรสจริง (PHYSTCAL ADDRESS GENERATION)

Memory ของ 8086-8088 จะเริ่มจาก address 0 ถึง FFFFFH ซึ่ง physical address จะมีทั้งหมด 20 bit จากลักษณะของ segmentation ของ 8086-8088 การอ้างถึง physical memory จะทำได้โดยการใช้แอดเดรสตั้งต้นของ segment (segment base address) ประกอบกับส่วนเพิ่ม (offset value) เช่น การ fetch instruction จาก code segment BIU จะใช้ค่าที่เก็บใน CS ซึ่งชี้ไปที่แอดเดรสตั้งต้น (base address) ของ code segment ประกอบค่าของ IP ซึ่งเป็น offset value บอกว่า instruction ที่จะ fetch อยู่ห่างจาก base address เท่าไร มาอ้างถึงphysical addressของinstruction ที่ต้องการ เราจะเห็นว่าทั้ง CS และ IP เป็น register ขนาด 16 bit ทั้งคู่ BIU จะใช้ค่าของ register ทั้งสองมาประกอบเป็น physical address ขนาด 20 bit ได้โดยวิธีที่แสดงได้ดังรูป คือ shift ค่าของ CS ไปทางซ้าย 4 bit แล้วนำมาบวกกับ offset คือค่าที่เก็บใน IP ประกอบกันเป็น 20-bit physical address ที่จะถูกส่งออกไปบน address bus ของ CPU เพื่อที่จะ fetch instruction มา execute

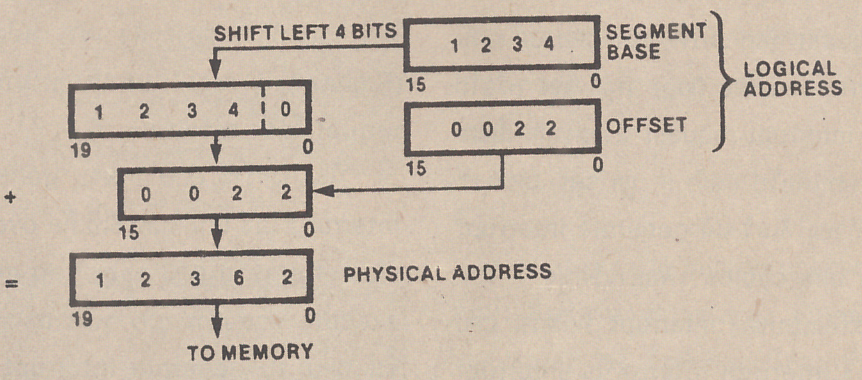
เราสามารถสรุปการทำ **physical address generation** ได้ดังนี้คือถ้าต้องการ **access memory** ใน **segment** ใด BIU จะเอาค่า **segment register** ซึ่งชี้ไปยัง **base address** ของ **segment** ดังกล่าวมา **shift** ไปทางซ้าย **4 bit** แล้วนำมาบวกกับ **offset**



รูปที่ 14 การจัดหน่วยความจำของ 8086/8088



รูปที่ 15 การจัดข้อมูลแบบต่าง ๆ ในหน่วยความจำ



รูปที่ 16 การกำหนด physical address

value *(ซึ่งอาจจะเป็นค่าที่เก็บใน* ***register*** *หรือค่า* ***offset*** *ภายในคำสั่ง)* กลายเป็น **physical address 20 bit** ตามต้องการ เนื่องมาจาก **offset value** มีขนาด **16 bit** ดังนั้น **segment** แต่ละ **segment** จึงมีขนาดสูงสุดได้ **64 Kbyte**

Memory ที่สงวนไว้ใช้สำหรับการทำงานภายในระบบ (DEDICATED AND RESERVED MEMORY)

บริเวณ address ต้น ๆ ท้ายสุดของ memory space จะถูกสงวนไว้สำหรับ interrupt system และ system reset โดย address บริเวณตั้งแต่ OH-3FFH จะถูกสงวนไว้เป็น interrupt pointer table ซึ่งจะ point ไปยังโปรแกรมที่จะทำงานเมื่อเกิดอินเตอรัพท์(interrupt service routine) ต่าง ๆ ส่วน memory address บริเวณ FFFFOH ถึง FFFFFH จะถูกสงวนไว้ใช้เมื่อเกิด SYSTEM RESET

ลักษณะการ access memory ของ 8086 และ 8088 มีข้อแตกต่างเล็กน้อย *8088 จะ* ***transfer*** *ข้อมูลผ่านทาง* ***data bus*** *ได้ทีละ* ***8 bit*** *เท่านั้น* แต่สำหรับ *8086 สามารถ* ***transfer*** *ได้ทีละ* ***16 bit*** *ถ้าคำสั่งอ้างถึง* ***word variable*** *ซึ่งตกอยู่บน* ***word boundary*** *(คือ* ***physical address*** *ที่เป็นเลขคู่) 8086 จะ* ***transfer data*** *ทีละ* ***16 bit*** *ได้ทันที แต่ถ้า* ***access*** *ถึง* ***word variable*** *ที่มี* ***Physical address*** *เป็นเลขคี่ 8086 จะทำการ* ***transfer*** *ทีละ* ***8 bit*** *2 ครั้ง ดังนั้นถ้าเราทำให้* ***word variable*** *ตกบน* ***word boundary*** *เสมอ จะทำให้ 8086 มี* ***performance*** *สูงขึ้น*

INPUT/OUTPUT

8086-8088 สามารถมี I/O space สำหรับต่อ external device ได้ถึง 64K 8-bit ports โดยมีคำสั่ง IN และ OUT ในการ transfer ข้อมูล ระหว่าง AL หรือ AX กับ device ภายนอก การ transfer ข้อมูลนี้สามารถทำได้ทีละ 8 bit หรือ 16 bit ก็ได้ 8088 จะ transfer ข้อมูล 16 bit ออกไปทีละ 8 bit ผ่าน data bus ในการทำงาน 2 cycle ของ bus แต่สำหรับ 8086 ถ้า I/O address เป็นเลขคู่ 8086 จะสามารถ transfer ออกไปทีละ 16 bit ในการทำเพียง 1 cycle ของ bus

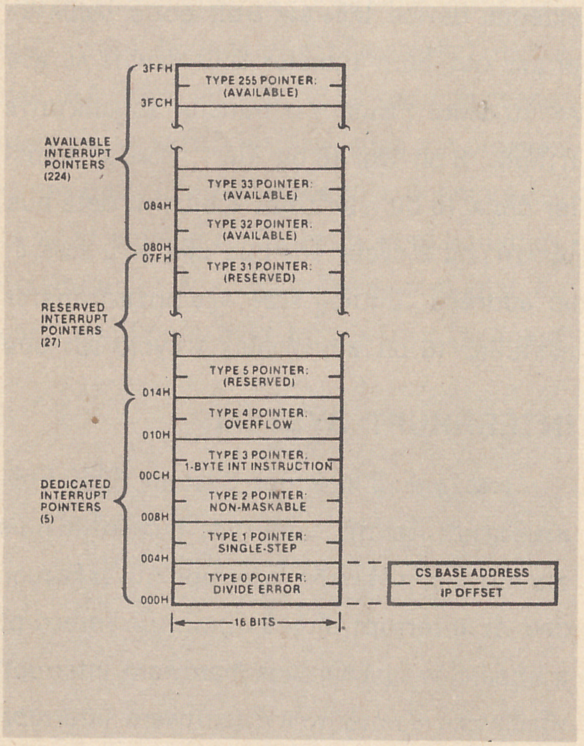
INTERRRUPT SYSTEM

8086/8088 มี Interrupt System ที่มีประสิทธิภาพซึ่งสามารถจัดการกับ interrupt ที่เกิดขึ้นได้ทั้งหมด 256 ชนิด โดยทุกครั้งที่เกิด interrupt นั้น จะต้องบอก CPU ว่า interrupt ที่เกิดขึ้นเป็นแบบใด interrupt สามารถเกิดได้จากอุปกณ์ภายนอก (hardware interrupt) หรือเกิดจากคำสั่งทางซอฟท์แวร์ (software interrupt) และอาจเกิดจากCPU เองที่ interrupt ในสภาวะของ

เงื่อนไขบางกรณี

external interrupt ที่เกิดจากฮาร์ดแวร์ภายนอก ส่งสัญญาณ request ที่ขา INTR ของ CPU CPU จะตรวจดู IF (interrupt-enable flag) ว่าถูก set หรือไม่ ถ้า IF ถูก clear interrupt request ดังกล่าวถูก disable CPU จะทำงานต่อไป แต่ถ้า IF ถูก set CPU จะทำคำสั่งที่กำลังทำอยู่ให้เสร็จแล้วgenerate interrupt acknowlaege bus cycles เพื่ออ่านข้อมูล 1 ไบท์จากบัสข้อมูล ซึ่งเป็นแบบของ interrupt ที่เกิดขึ้น CPU จะนำเอาแบบของ interrupt ที่ได้มาหาใน interrupt pointer table ซึ่งเป็นตารางที่เก็บ address ของ interrupt service routine ของ interrupt ในรูปแบบต่าง ๆ โดยเก็บในรูปของ CS และ IP ของ servic routine นั้น ๆ หลังจากที่ได้ CS,IP จากตารางแล้ว CPU จะเปลี่ยนข้อมูลที่บรรจุใน CS และ IP เพื่อไป execute ที่ service routine ตามต้องการ

คำสั่ง INT จะทำให้เกิด interrupt ภายในขึ้นหลังจากจบคำสั่ง โดยแบบของ interrupt จะถูกเก็บอยู่ใน opcode ของคำสั่งดังกล่าว จากนั้น CPU ก็จะเอาแบบ



รูปที่ 17 Interrupt Pointer Table

ของ interrupt ที่ได้ไปหาใน interrupt pointer table เพื่อหา address ของ interrupt service routine ต่อไป การทำงานของคำสั่ง INT เหมือนกับคำสั่ง RST (Restart) ใน 8080 แต่ INT สามารถมีแบบของ interrupt ได้ถึง 256 แบบ

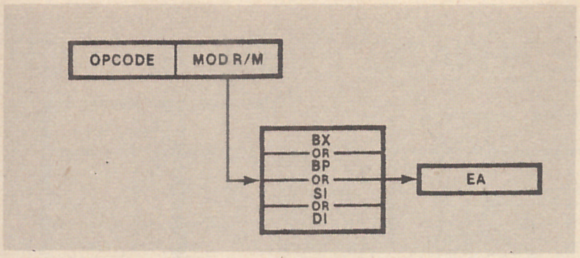
ในบางกรณี CPU อาจจะ generate internal interrupt เอง เช่น ในกรณีที่เกิด overflow เกิด device error หรือ CPU อยู่ใน trap mode เช่น ถ้าเรา execute คำสั่งหาร (DIV หรือ IDIV) แล้วเกิด error ขึ้น CPU จะ generate interrupt หลังจากกระทำคำสั่งดังกล่าวแล้ว โดยที่แบบของ interrupt ที่เกิดขึ้นจะเป็น type 0 ในกรณีที่ CPU อยู่ใน trap mode (user เป็นผู้ set IF เอง) CPU จะ generate interrupt แบบที่ 1 หลังจากที่ execute คำสั่งแต่ละคำสั่งเสร็จเสมอ

การอ้างแอดเดรส (ADDRESSING MODE) และชุดคำสั่ง (INSTRUCTION SET)

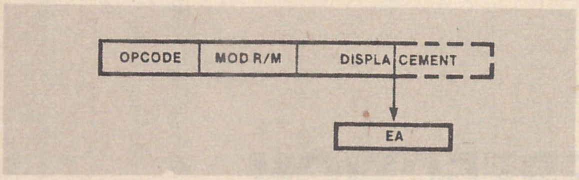
8088/8086 มี addressing mode ในการ access เอาค่า operand จาก Memory อย่างมีประสิทธิภาพหลายชนิด คือ

- ***Immediate and register indirect addressing***
- ***Direct addressing***
- ***Base Addressing***
- ***Indexed addressing***
- ***Based Indexed addressing***

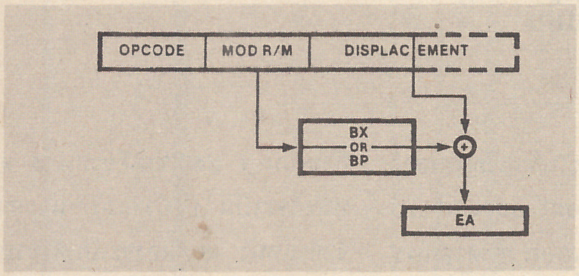
ซึ่งจาก addressing mode ดังกล่าว เราสามารถใช้ BX,BP เป็น based register และ SI กับ DI เป็น Index register ประกอบกับ displacement ซึ่งอยู่ในคำสั่ง opcode มารวมกันเป็น effective address (EA) ที่ต้องการจากลักษณะของ Addressing mode ที่มีประสิทธิภาพ ทำให้เราสามารถเขียนโปรแกรมได้ง่ายและสามารถ access ถึง operand ต่าง ๆ ได้เร็วขึ้น ชุดคำสั่งของ 8088/8086 มีคำสั่งที่มีประสิทธิภาพ เช่น คำสั่งคูณและหารเลข Sign หรือ Unsign binary Number ทั้ง 8 bit และ 16 bit คำสั่งเกี่ยวกับการทำ



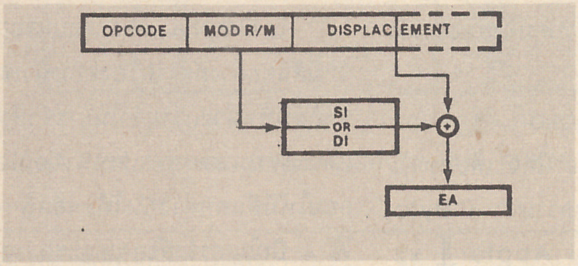
18.1 Register Indirect Addressing



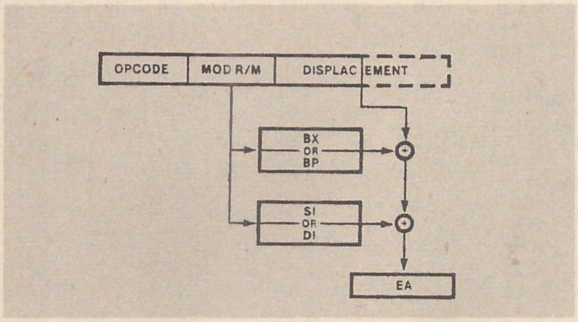
18.2 Direct Addressing



18.3 Base Addressing



18.4 Indexed Addressing



18.5 Based Indexed Addressing

รูปที่ 18 Addressing Mode แบบต่าง ๆ ของ 8086/8088

String Manipulation เช่น การ move,compare String ขนาดตั้งแต่ 1 ถึง 64 Kbyte คำสั่งเกี่ยวกับการ test bit ต่าง ๆ เป็นต้น สำหรับชุดคำสั่งและ addressing mode สามารถดูรายละเอียดได้จากหนังสืออ้างอิงข้างซ้าย

สรุป

8088/8086 เป็น **Microprocessor** ขนาด ***16 bit*** ที่มีประสิทธิภาพสูงมากตัวหนึ่ง ลักษณะเด่นที่น่าสนใจของ 8088/8086 คือ ลักษณะโครงสร้างสถาปัตยกรรมภายในที่เป็นลักษณะ **Pipelined Architecture** ทำให้การ **fetch** คำสั่งและการ **execute** คำสั่งสามารถทำได้พร้อม ๆ กัน เป็นผลให้ประสิทธิภาพการทำงานสูง การออกแบบ 8088/8086 ให้สามารถทำงานร่วมกับ **Coprocessor** คือ 8087,8089 ในระบบ **Multiprocessor System** ลักษณะของ **memory Segmentation** ซึ่งทำให้ง่ายต่อการเขียนโปรแกรม แบบ **modular programming** และสามารถพัฒนาไปใช้ในระบบ **memory management and protection** ซึ่งมีใช้ในเครื่อง **mini-computer** และ **mainframe** ชุดคำสั่งและ **addressing mode** ที่มีประสิทธิภาพของ 8088/8086 ทำให้ความเร็วและ **efficiency** ในการอ้างถึง **operand** สูงขึ้น จากลักษณะดังกล่าวมาข้างต้นมีผลทำให้ 8088/8086 เป็นไมโครโปรเซสเซอร์ที่มีประสิทธิภาพสูง และมีใช้แพร่หลายอยู่ในไมโครคอมพิวเตอร์หลายยี่ห้อ เช่น **IBM PC, Wang PC, Zenith, Future** ฯลฯ สำหรับ **Operating System** ที่ใช้กันมากกับเครื่องไมโครคอมพิวเตอร์เหล่านี้ คือ **MS-DOS** ของบริษัทไมโครซอฟท์และ **CP/M-86** ของ **Digital research** แนวโน้มการใช้งานของไมโครคอมพิวเตอร์ขนาด 16 บิท จะมีมากขึ้นเป็นลำดับ และมีแนวโน้มจะเข้าแทนที่ไมโครคอมพิวเตอร์ขนาด 8 บิทที่ใช้กันอยู่ต่อไป

Ø

หนังสืออ้างอิง

1. **iPAX 88** ของ **Intel Comperation**

2 **iPAX 88 user manual** ของ **Intel Coperation**

3. **8086 book** ของสำนักพิมพ์ **Osborne**

POKING PERFECTION

อาร์ทีเมีย

เราไม่สามารถใช้คำสั่ง HLIN, VLIN และ PLOT ใน mode TEXT screen ของ Apple เพราะว่าคู่มือไม่ได้อธิบายไว้ แต่มันก็ยังเป็นไปได้

คำสั่ง HLIN, VLIN และ PLOT ใน Low-resolution graphic นี้จะทำหน้าที่ลากเส้นในแนวระดับ แนวตั้ง และสร้างจุดสี่เหลี่ยมเล็ก ๆ ที่จอภาพ คำสั่งเหล่านี้เรายังสามารถใช้ได้ใน mode TEXT หรือ mode ของตัวอักษรอีกด้วยทั้งด้วยภาษา BASIC หรือภาษาเครื่อง ในบทความนี้คุณจะได้ทราบถึงการเก็บข้อมูลลงหน่วยความจำ ในการทำเกี่ยวกับ Low-resolutian graphic ของ Apple Ⅱ

คำสั่ง GR จะเปลี่ยนจอภาพของ Apple Ⅱ จาก Text screen ไปเป็น graphic screen ซึ่งตัวอักษรแต่ละตัวจะถูกแบ่งออกเป็นสองส่วน (ดูรูปที่ 1) หน่วยความจำ แอดเดรส 48 จะเก็บข้อมูลซึ่งควบคุมทั้งสองส่วนนี้ (ดูรูปที่ 2)

ถ้ามีคำสั่ง COLOR = 1 หน่วยความจำตำแหน่ง 48 นี้ก็จะเก็บค่าเท่ากับ 17 คือในหน่วยความจำ 1 Byte หรือ 1 หน่วยของ Apple Ⅱ จะประกอบด้วย 8 Bit หรือ 8 ส่วน คือเป็นเลขฐานสอง 8 หลัก 4 หลักแรกจะคุม Graphic ช่วงล่างและ 4 หลักหลังจะคุม Graphic ช่วงบน เมื่อเราสั่ง COLOR = 1 4หลักแรกก็จะเป็น 1 และเช่นเดียวกัน 4 หลักหลังก็จะเป็น 1 ด้วยและหน่วยความจำตำแหน่ง 48 นี้เองจะเป็นตัวควบคุมสีของจุดต่าง ๆ บนจอภาพใน mode Graphic

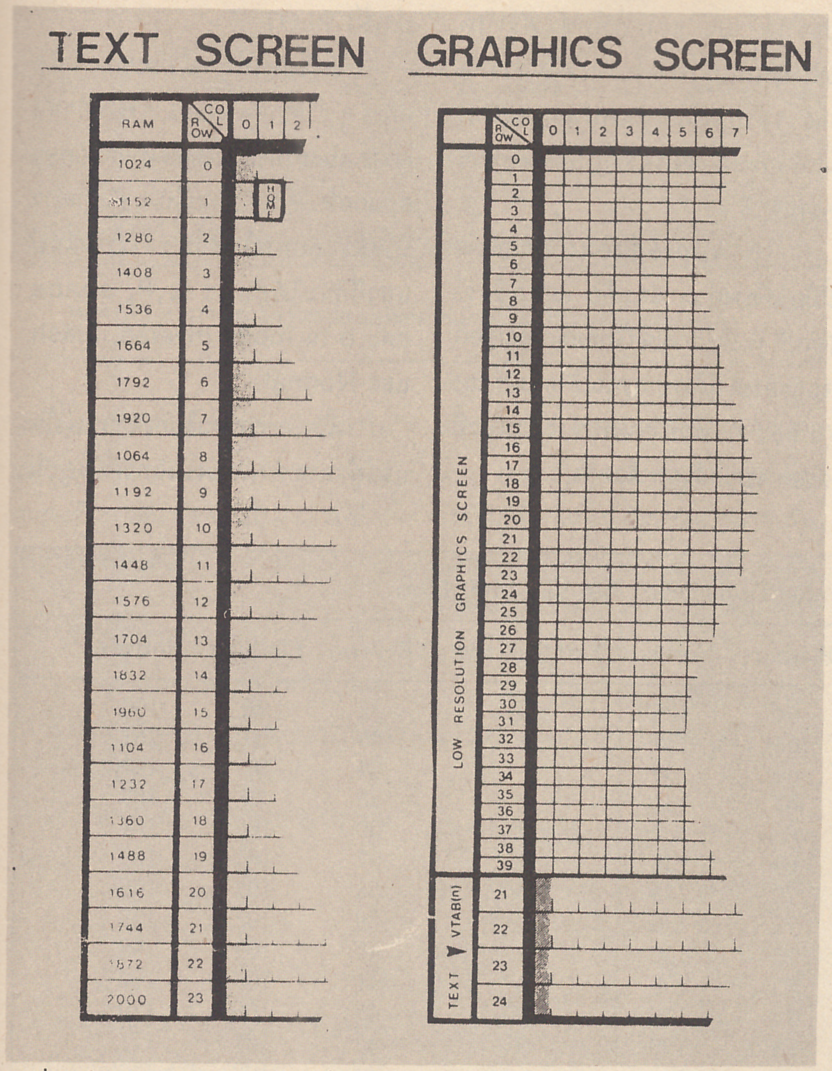
ใน Graphic แต่ละจุดบนจอภาพจะมีสีแตกต่างกัน เครื่องคอมพิวเตอร์จะต้องใช้หน่วยความจำในการจำสีของแต่ละจุดบนจอภาพ และใน Apple Ⅱ ย่อมให้มีสีแตกต่างกันได้ 16 สี คือใช้ 4 บิทในการจำสีของแต่ละชุด หน่วยความจำ 1 หน่วยหรือ 1 Byte นี้ จึงจำค่าสีของจุด 2 จุด และนี่คือเหตุผลที่

```
100 HOME
110 FOR K = 0 TO 255
120 VTAB 10: HTAB 5
130 PRINT "VALUE POKED INTO BYTE 48 = ";K;" = "
140 POKE 48,K
150 VLIN 0,47 AT 36
200 GET A$
210 NEXT K
```

โปรแกรมที่ 1 โปรแกรมสั้น ๆ ภาษา BASIC แสดงผลของคำสั่ง Graphic บน Text screen

```
100 HOME
110 FOR K = 0 TO 255
120 POKE 48,K
130 FOR J = 0 TO 39 STEP 2
140 PLOT J,J
150 PLOT J,J + 1
160 NEXT J
170 FOR J = 39 TO 0 STEP - 2
180 PLOT 39 - J,J
190 PLOT 39 - J,J + 1
200 NEXT J
300 GET A$
310 NEXT K
```

โปรแกรมที่ 2 พยายามที่จะใช้โปรแกรมภาษา BASIC สร้างตัวอักษรจากตารางที่ 1



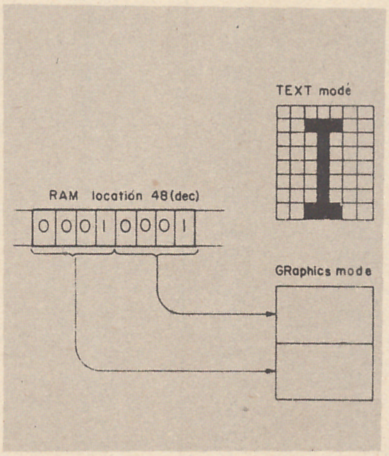
รูปที่ 1 เมื่อ **Text screen** ถูกเปลี่ยนไปเป็น **Graphic mode** แต่ละตัวอักษร จะถูกแบ่งครึ่งเป็น 2 ส่วน

Graphic screen ละเอียดกว่า Text screen ถึง 2 เท่า

คำสั่ง VLIN

หน่วยความจำตำแหน่ง 48 นี้สามารถทำให้มีค่าได้ตั้งแต่ 0 ถึง 255 ซึ่งจะให้ผลลัพธ์ที่น่าสนใจใน mode Graphic ยิ่งใน mode Text ด้วย แล้วมีอะไรที่น่าสนใจมากกว่า ลองคีย์โปรแกรมสั้น ๆ ในโปรแกรมที่ 1 แล้วดูผลของมัน

ตารางที่ 1 มีรายละเอียดของตัวอักษรและ mode Reverse, Flash และ Normal) ของมัน คำสั่ง VLIN



รูปที่ 2 ตัวอักษรในจอภาพจะถูกควบคุมโดยหน่วยความจำตำแหน่ง 48 เมื่ออยู่ใน **Graphic mode**

จะแสดงผลบนจอภาพของ Apple II ใน mode Text

คำสั่ง HLIN

จากนั้นก็เพิ่มบรรทัด 160 ลงไปในโปรแกรมที่ 1

160 HLIN 0,39 AT 22

เมื่อคุณ RUN โปรแกรมนี้ คุณก็จะได้เห็นตัวอักษรในแนวตั้งมาปรากฏอีกครั้งในแนวนอนแต่จะต่างกัน

กลับมาที่หน่วยความจำตำแหน่ง 48 อย่างที่กล่าวมาแล้ว 4 บิทแรกจะควบคุมครึ่งล่างของตัวอักษร และ 4 บิทหลังจะควบคุมครึ่งบนของตัวอักษร (ดูรูปที่ 2) ถ้าจอภาพถูกแบ่งออกเป็น 48 แถวมีเบอร์กำกับเป็น 0 ถึง 47 แล้ว 4 บิทหลังของหน่วยความจำตำแหน่ง 48 นี้ก็จะควบคุมแถวซึ่งมีเบอร์เป็นเลขคู่ คือ 0,2,4,6,....,46 และ 4 บิทแรกก็จะควบคุมแถวซึ่งเป็นเลขคี่คือแถว 1,3,5,7,.......,47

เพื่อที่จะเข้าใจยิ่งขึ้น คีย์บรรทัด 170 นี้เพิ่มเข้าไปในโปรแกรมที่ 1

170 HLIN 0,39 AT 25

เมื่อคุณ RUN โปรแกรมอีกครั้ง คุณก็จะพบว่า บรรทัด 160 ซึ่งแสดงผลในแถว 22 ซึ่งเป็นแถวคู่นั้น จะให้ผลเป็นอักษรตามลำดับดังนี้

space,!,",#,$,%,&,',
(,),*,+,,,–,.,/

ซึ่งจะเปลี่ยนทุก ๆ ครั้งที่หน่วยความจำที่ 48 นี่เปลี่ยนไป แต่ในบรรทัด 170 ซึ่งแสดงผลที่แถว 25 ซึ่งเป็นแถวคี่นั้นจะให้ลำดับของตัวอักษรดังนี้

@, P, space,O

ในรูปของ Inverse, Flash และ Normal แต่จะเปลี่ยนทุก ๆ ครั้งที่ค่าของหน่วยความจำตำแหน่ง 48 นี้เป็นจำนวนเท่าของ 16

จากนี้ก็เพิ่ม 2 บรรทัดสุดท้ายเข้าไปในโปรแกรมนี้

180 HLIN 0,38 AT 30
190 HLIN 0,39 AT 31

คราวนี้เมื่อ RUN โปรแกรมแล้ว จะเห็นว่าผลลัพธ์ที่ได้จากสองบรรทัดนี้จะได้ผลลัพธ์เพียงบรรทัดเดียวบนจอภาพ และเหมือนกับผลที่ได้จากคำสั่ง VLIN

สังเกตดูในบรรทัด 180 และ 190 เราลากเส้นด้วยคำสั่ง HLIN ในแถวคู่ 30 และแถวคี่ 31 ทั้งสองคำสั่งถูกกระทำในเวลาที่ต่างกันเพียงเล็กน้อย ในบรรทัด 180 จะทำให้ผลลัพธ์เป็นอย่างเช่นในบรรทัด 160 แล้วในบรรทัด 190 จะไปแก้ไขค่าในหน่วยความจำที่จะออกจอภาพให้ตรงกับค่าในหน่วยความจำที่ 48

ใน Apple][text screen นี้มีอยู่เพียงแค่ 24 แถว คือ แถว 0 ถึง 23 (ดูรูปที่ 1) ซึ่งแถวคู่ใน Graphic screen จะถูกอ้างถึงได้โดยการคูณด้วยเบอร์แถว (0 ถึง 23) ด้วย 2 และอ้างถึงแถวคี่ด้วยการบวกด้วย 1 อีกครั้ง

คำสั่ง PLOT

การใช้งานคำสั่ง PLOT นี้ก็คล้าย ๆ กับคำสั่ง HLIN การใช้คำสั่ง PLOT ในแถวคู่จะทำให้เกิดเป็นตัวอักษร space!" # $....และอื่น ๆ ดังที่กล่าวมาแล้วและการใช้คำสั่ง PLOT นี้กับแถวคี่ก็จะได้อักษร @,P, space และ 0 ใน mode Inverse, Flash และ Normal

เมื่อ PLOT ลงไปยังตำแหน่งในแถวคู่ และตำแหน่งในแถวคี่ซึ่งคู่กัน

Value To Be Poked into Video RAM for:							
Text Character	Reverse	Flashing	Normal	Text Character	Reverse	Flashing	Normal
@	0	64	128	space	32	96	160
A	1	65	129	!	33	97	161
B	2	66	130	"	34	98	162
C	3	67	131	#	35	99	163
D	4	68	132	$	36	100	164
E	5	69	133	%	37	101	165
F	6	70	134	&	38	102	166
G	7	71	135	'	39	103	167
H	8	72	136	(	40	104	168
I	9	73	137	)	41	105	169
J	10	74	138	*	42	106	170
K	11	75	139	+	43	107	171
L	12	76	140	,	44	108	172
M	13	77	141	–	45	109	173
N	14	78	142	.	46	110	174
O	15	79	143	/	47	111	175
P	16	80	144	0	48	112	176
Q	17	81	145	1	49	113	177
R	18	82	146	2	50	114	178
S	19	83	147	3	51	115	179
T	20	84	148	4	52	116	180
U	21	85	149	5	53	117	181
V	22	86	150	6	54	118	182
W	23	87	151	7	55	119	183
X	24	88	152	8	56	120	184
Y	25	89	153	9	57	121	185
Z	26	90	154	:	58	122	186
[	27	91	155	;	59	123	187
\	28	92	156	<	60	124	188
]	29	93	157	=	61	125	189
^	30	94	158	>	62	126	190
—	31	95	159	?	63	127	191

ตารางที่ 1 ค่าต่าง ๆ ที่กำหนดให้หน่วยความจำตำแหน่ง 48 จะถูกแสดงเป็นตัวอักษรใน Text screen

Routine	Entry Point	Accumulator	Y Register	Other RAM to Set
VLIN	$F828	high row	column	low row in byte 45
HLIN	$F819	row number	left column	right column in byte 44
PLOT	$F800	row number	column	—

ตารางที่ 2 ตำแหน่งต่าง ๆ ที่จะต้องติดต่อด้วยเมื่อจะใช้โปรแกรมย่อยของเครื่อง

```
100 FOR K = 0 TO 255
110 POKE 48,K
120 HLIN 39,0 AT 0
130 HLIN 39,0 AT 1
140 VLIN 0,47 AT 0
150 HLIN 0,39 AT 46
160 HLIN 0,39 AT 47
170 VLIN 47,0 AT 39
180 GET A$
190 NEXT K
```

โปรแกรมที่ 3 จะลากเส้นรอบจอภาพ

ตัวอักษรที่ได้ก็จะตรงกับค่าที่เก็บไว้ในหน่วยความจำตำแหน่ง 48 ตามค่าในตารางที่ 1 ลองทดสอบโปรแกรมที่ 2 เพื่อเข้าใจเรื่องนี้

POKE ที่จอภาพ

คุณสามารถ POKE หรือกำหนดค่า ซึ่งแสดงอยู่ในตารางที่ 1 ไปยังหน่วยความจำที่ใช้ออกจอภาพได้เลย เพื่อทดลองเกี่ยวกับเรื่องนี้ คีย์อีก 2 บรรทัดเพิ่มเติมในโปรแกรมที่ 2

200 POKE 1299, K
210 POKE 1300, K

ในรูปที่ 1 จะแสดงค่าของตำแหน่งในหน่วยความจำ ซึ่งเป็นคอลัมน์แรกในแต่ละบรรทัดของ text screen สำหรับในคอลัมน์อื่น ๆ ให้บวกค่าของคอลัมน์ (0 ถึง 39) เข้าไปในค่าซึ่งให้มาในรูปที่ 1 นั้น

คำสั่ง VLIN, HLIN และ PLOT จะเรียกใช้โปรแกรมย่อยซึ่งเป็นภาษาเครื่อง ซึ่งจะให้ผลลัพธ์ที่เร็วกว่าในภาษา BASIC ซึ่งทำหน้าที่เหมือนกัน คุณจะพิสูจน์ได้จากโปรแกรมที่ 3 ซึ่งทำหน้าที่ลากเส้นรอบจอภาพ ด้วยค่าต่าง ๆ ตั้งแต่ 0 ถึง 255 จากนั้นก็ทดลองเขียนเป็นภาษา BASIC แบบธรรมดา ซึ่งจะทำหน้าที่เช่นเดียวกับโปรแกรมที่ 3

การใช้โปรแกรมย่อยของเครื่อง

การโปรแกรมด้วยภาษาเครื่องสามารถเรียกใช้ คำสั่ง HLIN, VLIN และ PLOT ได้โดยโหลดค่าซึ่งจะเป็นจุดเริ่มต้น จุดสิ้นสุดของการลากเส้น และตำแหน่งแถวหรือคอลัมน์ของเส้นที่จะลาก แล้วก็เรียกใช้ไปยังตำแหน่งของโปรแกรมย่อยนั้น ๆ

ในตารางที่ 2 จะแสดงถึงตำแหน่งต่าง ๆ ที่คุณต้องใช้เพื่อติดต่อกับโปรแกรมย่อยเหล่านี้ โปรแกรมที่ 4 จะแสดงถึงวิธีการติดต่อกับตัวอย่างเหล่านี้

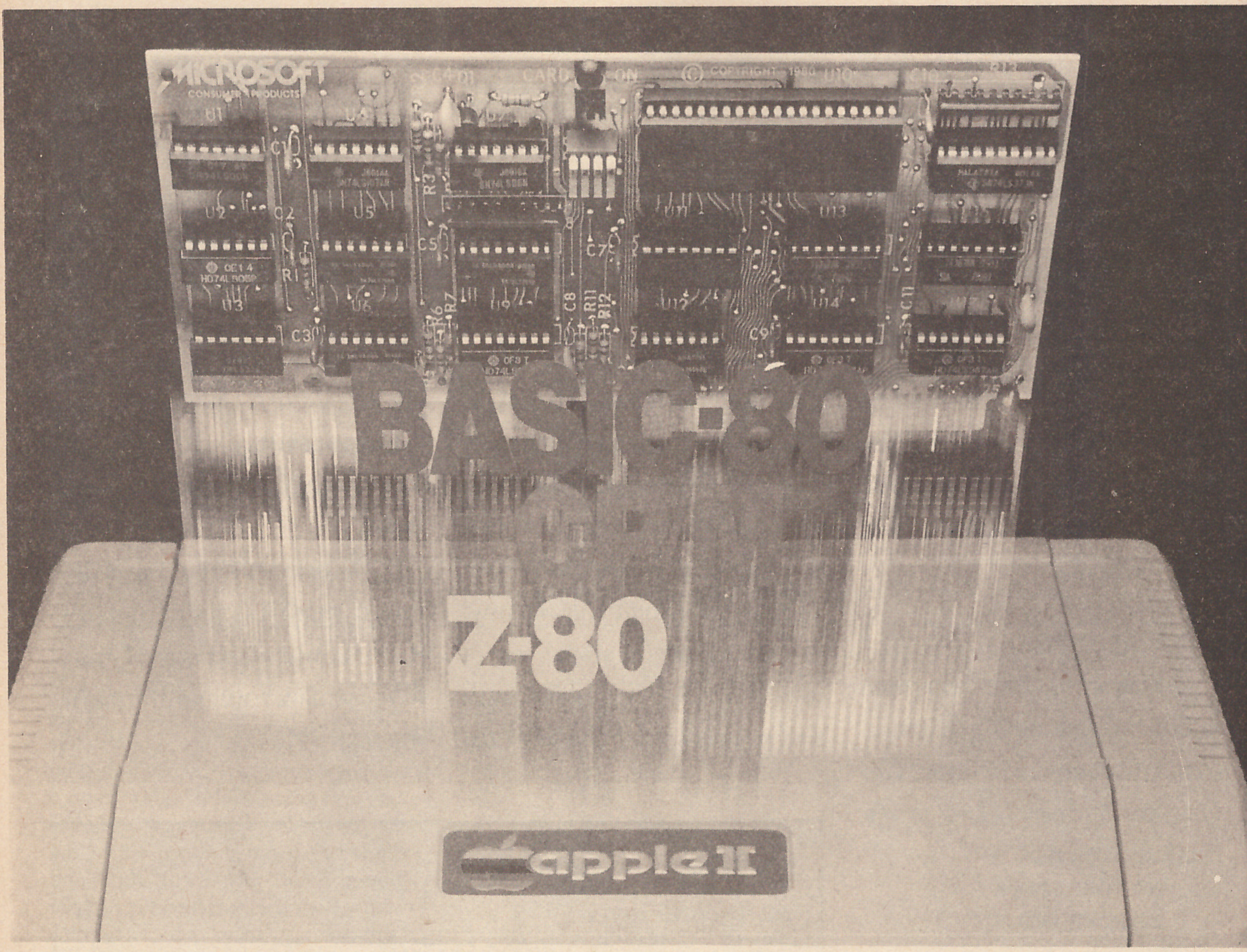
ทั้งในภาษาเครื่อง และภาษา BASIC คุณต้องลากเส้นด้วยคำสั่ง HLIN ทั้งในแถวคู่และในแถวคี่ตัดมา จึงจะให้ตัวอักษรเหมือนกับที่ลากเส้นด้วย VILN ∅

```
PRESS ANY KEY TO CONTINUE

SOURCE FILE: H/V B
---- NEXT OBJECT FILE NAME IS H/V B.OBJ0
6000:                 ORG  $6000   ;ORIGIN
6000:20 58 FC         JSR  $FC58   ;HOMER
6003:20 35 FD    CHAR JSR  $FD35   ;GET CHAR
6006:85 30            STA  $30     ;STORE 48
6008:A9 2F            LDA  #47     ;STOP ROW
600A:85 2D            STA  $2D     ;STORE 45
600C:A0 00            LDY  #0      ;COLUMN
600E:A9 00            LDA  #0      ;START ROW
6010:20 28 F8         JSR  $F828   ;VLIN
6013:A0 27            LDY  #39     ;COLUMN
6015:A9 00            LDA  #0      ;START ROW
6017:20 28 F8         JSR  $F828   ;VLIN
601A:A9 27            LDA  #39     ;STOP COLUMN
601C:85 2C            STA  $2C     ;STORE 44
601E:A0 00            LDY  #0      ;START COL
6020:A9 00            LDA  #0      ;ROW EVEN #
6022:20 19 F8         JSR  $F819   ;HLIN
6025:A0 00            LDY  #0      ;START COLUMN
6027:A9 01            LDA  #1      ;ROW ODD (EVEN #+1)
6029:20 19 F8         JSR  $F819   ;HLIN
602C:A0 00            LDY  #0      ;START COLUMN
602E:A9 2E            LDA  #46     ;ROW EVEN #
6030:20 19 F8         JSR  $F819   ;HLIN
6033:A0 00            LDY  #0      ;START COLUMN
6035:A9 2F            LDA  #47     ;ROW ODD (EVEN #+1)
6037:20 19 F8         JSR  $F819   ;HLIN
603A:4C 03 60         JMP  CHAR    ;DO ANOTHER CHARACTER

*** SUCCESSFUL ASSEMBLY: NO ERRORS
```

โปรแกรมที่ 4 โปรแกรมภาพเครื่องซึ่งเรียกใช้ HLIN และ VLIN เพื่อลากกรอบภาพ



MBASIC : REFERENCE

สัญลักษณ์และคำย่อ

exp - นิพจน์ทางตรรกศาสตร์ (logic), string หรือคณิตศาสตร์ (arithmatic) ซึ่งอาจเป็น integer, single precision หรือ *double precision*

string - ชุดของตัวอักษร

null string - string ที่ยาว = 0 คือไม่มีอะไรเลย

< > - บอกว่าข้อความใน < > เป็นชื่อคีย์ที่ต้องกด

[ ] - ข้อความตามรูปแบบในวงเล็บจะใส่หรือไม่ก็ได้ แต่ไม่ต้องใส่ตัววงเล็บนี้

( ) - วงเล็บนี้ต้องใช้ตามที่แสดงไว้

dummy argument - ข้อมูลที่ให้แก่ฟังก์ชั่น แต่ไม่ได้ถูกนำไปใช้ในการทำงานของฟังก์ชั่น

default - ค่าที่ตั้งไว้ในระบบ ถ้าไม่มีการแก้ไขเป็นอย่างอื่นจะถือเอาค่านี้ในการทำงาน

Capital letter - อักษรตัวใหญ่เป็น keyword ต้องใช้ตามที่แสดงไว้

การใช้คำสั่งใน BASIC - 80

แบ่งเป็น 2 level คือ...

- ***Command level*** โดยการป้อนคำสั่งตามหลัง prompt string "OK" (ไม่มี line number นำ) คำสั่งจะถูกทำตามทันที และกลับสู่ command level (บอก "OK") อีกเมื่อเสร็จ
- ***Program level*** โดยการป้อนคำสั่งในโปรแกรม (มี line number นำหน้า) BASIC - 80 จะทำตามคำสั่งดังกล่าวไปตามลำดับจนกว่าจะจบโปรแกรมจึงจะกลับสู่ command level (บอก "OK")

การตั้งชื่อต่าง ๆ

- ชื่อตัวแปรยาวได้ถึง 100 ตัวอักษร และมีความสำคัญทุกตัว

- ตัวแปรเก็บ String มี $ ต่อท้ายชื่อ

- ชื่อ file ตามข้อกำหนดของโปรแกรมควบคุมระบบที่ใช้ (ใน CP/M ใช้ไม่เกิน 8 ตัวอักษร และ file extension อีก 3 ตัวหลังจุด)

ลำดับการทำงานตามเครื่องหมายต่าง ๆ

จากก่อนไปหลัง	*สัญลักษณ์*
1. นิพจน์ในวงเล็บ	( )
2. ยกกำลัง	^
3. นิเสธ	-exp
4. การคูณ,หารแบบ floating point	* , /
5. การหารตัดเศษ	\ (backslash)
6. การหารเอาเฉพาะเศษ	MOD
7. การบวก,ลบ	+ -
8. Relational operator	=,< >,< >,=,=
9. Not	NOT
10. And	AND
11. Or	OR
12. Exclusive Or	XOR
13. Implied	IMP
14. Equivalence	EQV

BASIC-80 COMMAND

AUTO [line [increment]] พิมพ์ line number เริ่มตั้งแต่ line เพิ่มทีละ increment

CLEAR [,[exp1] [,exp2]]

ลบค่าตัวแปรที่เก็บตัวเลขเป็นศูนย์ ตัวแปรเก็บ string เป็น null string และ set ค่าสูงสุดที่ใช้ได้ในหน่วยความจำเป็น exp1 และขนาดของ stack เป็น exp2

CLOAD [?]filename

ใช้เหมือน LOAD แต่ load จากเทป cassette (มีเฉพาะ version ที่ใช้กับเทป) ถ้ามี ? จะเป็นการตรวจดูข้อมูลในเทปกับในหน่วยความจำ ถ้าตรงกันจะพิมพ์ OK ถ้าไม่จะบอก NO good

CSAVE filename

ใช้เหมือน SAVE แต่ save ลงในเทป cassette (มีเฉพาะ version ที่ใช้กับเทป)

CONT ให้โปรแกรมทำงานต่อหลังจากหยุดลงด้วยคำสั่ง STOP,END หรือ Control-C

DELETE [starting line], [-ending line](ใน apple version ย่อได้เป็น DEL)

EDIT LINE NUMBER เข้าสู่ Edit mode ที่สามารถแก้ไขโปรแกรม line นั้น ๆ ได้ มีคำสั่งย่อยคือ

A - เริ่มการ edit ใหม่ที่ต้น line โปรแกรมเดิมยังไม่เปลี่ยนแปลง

[n]C string - เปลี่ยนตัวอักษรถัดไป n ตัวเป็น String ยาว n ตัว

[n]D - ลบตัวอักษรถัดไป n ตัว

E หรือ<Return>= ออกจาก Edit mode และใช้คำตอบที่แก้

H string <esc> = ลบส่วนถัดไปทั้งหมดของ line, แทรก string แล้วเลิกการแทรก

I string<esc> - แทรก string แล้วเลิกการแทรก

[n] K c - ลบส่วนของ line ตั้งแต่นี้จนถึงอักษร "C" ตัวที่ n

L - พิมพ์ส่วนที่เหลือของ line และกลับมาเริ่ม edit mode ใหม่ที่ต้นบรรทัด

Q - เลิก edit และไม่แก้ไขโปรแกรมเดิม

[n]S C - หาอักษร "C" ตัวที่ n

X string<esc>- พิมพ์บรรทัดจนจบ และเริ่มแทรก String ต่อท้าย

n <←—> - เลื่อน cursor ถอยหลังไป n ตัว

n<Space Bar> พิมพ์ตัวอักษรถัดไป n ตัว

FILES [filename] พิมพ์รายชื่อ file บนดิสค์แผ่นที่ใช้อยู่ซึ่งสอดคล้องกับ filename ที่กำหนด (อาจใช้ ? แทนอักษรใด ๆ ในชื่อหรือ extension และ * แทนชื่อหรือ extension ใด ๆ)

LIST [line number [-[line number]]]

พิมพ์โปรแกรมทั้งหมด, เฉพาะ line หรือเฉพาะในช่วงที่กำหนดออกทางจอ

LLIST ใช้เหมือน List แต่พิมพ์อก printer แทน

LOAD filename [,R]

Load โปรแกรมจาก file ในดิสค์, R หมายถึงให้ RUN เลยโดยไม่ได้ไฟล์เดิม

MERGE filename

Load โปรแกรมจาก file ในดิสค์ (ต้องเก็บเป็นรหัส ASC II เข้ามารวม (merge) กับโปรแกรมเดิมในหน่วยความจำ

NAME oldname AS newname

เปลี่ยนชื่อ file บนดิสค์จาก oldname เป็น newname

NEW ลบโปรแกรมและตัวแปรเดิมในหน่วยความจำทั้งหมด

NULL exp ตั้งจำนวน null character ที่จะพิมพ์ต่อท้ายแต่ละบรรทัด **(ไม่มีใน Apple version)**

RENUM [[new line]], old line] [,increment]]]
จัด line number ใหม่ตั้งแต่ old line เป็นต้น เปลี่ยนเป็นเริ่มตั้งแต่new lineและเพิ่มทีละ increment

RESET ทำการ RESET CP/M หลังจากการเปลี่ยนดิสค์

RUN [line number] execute โปรแกรมเริ่มตั้งแต่ต้นหรือตั้งแต่ line number

RUN filename [,R] จัดการ Load โปรแกรมจาก file บนดิสค์ และ RUN ส่วน, R ทำให้ file ที่เปิดก่อน RUN ยังคงเปิดอยู่

SAVE filename [,A or, P] จัดการ Save โปรแกรมในหน่วยความจำลงใน file บน ดิสค์ ส่วน, A ทำให้ save เป็นรหัส ASC II และ ,P ทำให้ file ถูก protect คือ list ไม่ได้

SYSTEM ปิด file ทั้งหมดและกลับสู่ CP/M

TRON/TROFF เริ่ม/เลิก การติดตามลำดับการทำงาน ของโปร่แกรม **(ใน Apple Version ใช้ TRACE/NOTRACE แทน)**

WIDTH [LPRINT] exp จัดความกว้างของ line ที่จะแสดงบนจอ (หรือบน printer เมื่อใช้ LPRINT) ให้กว้างเท่ากับค่าของ exp (อยู่ในช่วง 15 -255 และค่า default เป็น 72) **(Apple version มีค่า default เป็น 40 หรือ 80)**

BASIC-80 STATEMENT

CALL VARIABLE [(argument list)]
เรียกใช้โปรแกรมย่อยภาษา Assembly ซึ่งมีแอดเดรสตั้งต้นตามค่าใน variable

CHAIN [MERGE]filename [,[line number exp] [,DELETE RANGE]
ส่งการควบคุมให้โปรแกรมใน file ที่กำหนดชื่อ ให้ทำงานต่อ ส่วน option ต่าง ๆ คือ
-MERGE รวมโปรแกรมที่ CHAIN เข้ากับโปรแกรมเดิม (ทำ overlay)
-line exp กำหนดจุดที่จะเริ่มทำงานตามโปรแกรมที่รวบเข้ามา ถ้าไม่มีจะทำตั้งแต่บรรทัดแรก
-ALL จัดการส่งผ่าน parameter ไปยังโปรแกรมใหม่ที่ CMAIN เข้ามา โดยไม่ใช้ COMMON
-DELETE range ลบบรรทัดที่ไม่ต้องการในช่วง range ออก

CLOSE [list of [#] file number]
ปิด file number ถ้าไม่กำหนดจะปิดทุกไฟล์

COMMON list of variable ส่งค่าตัวแปรใน list ไปยังโปรแกรมที่ CHAIN เข้ามา

DATA list of constant เก็บข้อมูลทั้งตัวเลขและ string สำหรับอ่านโดยคำสั่ง read

DEF FName [(parameter list) + function definition
กำหนดฟังก์ชันทางคณิตศาสตร์หรือ string ที่เรียกใช้ได้โดย name [(parameter list)]

DEF **type** range of letters
กำหนดตัวแปรที่ขึ้นต้นด้วย

ตัวอักษรใน range(s) ให้มี type เป็น integer (DEFINT), single precision (DEFSNG), double precision (DEFDBL) หรือ string (DEFSTR)

DEF USR [n] + address — กำหนดแอดเดรสตั้งต้นของโปรแกรมย่อยภาษา Assembly ที่ n (n = 0-9 และ default เป็น 0) address อาจเป็น integer exp ได้

DIM list of subscripted variable

กำหนดว่าตัวแปรใดจะเป็นตัวแปรแบบ array กี่มิติและมีขนาดสูงสุดเท่าไร โดยใช้ variable name (maximum subscript)

END — หยุดการทำงานตามโปรแกรม ปิดไฟล์ทั้งหมดและกลับสู่ BASIC Command level

ERASE list of array variables

ลบตัวแปรแบบ array ที่ DIM ไว้ ออกจากโปรแกรม

ERROR Code — สร้าง error code (code = 0-255) ตัวแปร ERR จะเก็บ error code และ ERL จะเก็บค่า line ที่เกิด error เพื่อใช้ในโปรแกรมจัดการกับความผิดพลาด (error trapping routine)

FIELD [#] file number field width AS string variable [,field width AS string variable...]

จัดที่ใน buffer สำหรับ random file

FOR variable = exp 1 TO exp2 [STEP exp3]

สร้าง loop ที่เริ่มตั้งแต่ variable เท่ากับ exp1 จนถึง exp2 เพิ่มค่าทีละ exp3 (default เป็น 1)

GET [#] file number [,record number]

อ่าน record จาก random file (ถ้าไม่กำหนด record number จะอ่าน record ถัดจากเดิม)

GOSUB line number — ไปทำงานตามโปรแกรมย่อยที่ line number

GOTO line number — ข้ามไปทำงานที่ line number

IF exp THEN statement [:statement...] [ELSE statement หรือ line namber]

ถ้า exp มีค่าไม่เท่ากับ 0 จะทำตาม statement หลัง then ไม่เช่นนั้นจะทำตาม statement หลัง else (หรือไปทำที่ line number หลัง ELSE) หรือ statement ถัดไป

IF exp **THEN** line number **[ELSE** statement หรือ line number]

IF ixp **GOTO** line number **[ELSE** statement หรือ line number]

ถ้า exp มีค่าไม่เท่ากับ 0 จะไปทำงานที่ line number ไม่เช่นนั้นจะทำตาม statemet หลัง else (หรือไปทำที่ line number หลัง ELSE) หรือ statement ถัดไป

INPUT [;] ["prompt string";] list fo variable

อ่านข้อมูลจากคีย์บอร์ด semicolon หลัง INPUT จะทำให้ไม่มีการขึ้นบรรทัดใหม่บนจอเมื่อป้อนข้อมูลจบและกด RETURN ส่วน semicolon หลัง prompt string จะทำให้มีการพิมพ์เครื่องหมายคำถาม ถ้าใช้ comma แทนก็จะไม่พิมพ์

INPUT # file number, list of variable
อ่านข้อมูลจาก Sequential disk file

KILL filename — ลบ file ทิ้งจาก disk

LET variable= exp — กำหนดค่าจาก exp ให้ variable (คำว่า LET อาจละได้)

LINE INPUT [;] ["prompt string";] string variable
อ่าน string จากคีย์บอร์ด semicolon หลัง LINE INPUT ทำให้ไม่ขึ้นบรรทัดใหม่เมื่อกด Return

LINE INPUT #File number, string variable
อ่าน string จาก sequential disk file

LPRINT [USING 'format string';] list of exp
พิมพ์ข้อมูลออกทาง line printer (รายละเอียดเหมือน PRINT)

LSET string variable = string exp
จัด string เข้าใน buffer ของ random file โดยชิดซ้าย

MID$ (string exp1,n[,m].string exp1=string exp2
ตั้งแต่ตำแหน่ง ด้วย string exp2 (n และ m ที่ n ไป m ตัวจะถูกแทนที่เป็น integer exp) ถ้าไม่ใส่ m จะใช้ string exp2 ทั้งหมด

NEXT [list of variable] — บอกตำแหน่งสิ้นสุด FOR/NEXT LOOP

ON ERROR GOTO line number
บอกตำแหน่งของ โปรแกรมจัดการกับความผิดพลาด (error trapping routine) ที่ไปทำเมื่อเกิด error ขึ้น

ON EXP GOSUBlist of line number
ไปทำงานตามโปรแกรมย่อยที่ line number ตัวที่ n ใน list (หลังค่า integer ที่ได้จาก exp แล้วปัดเศษ

ON exp GOTO list of line number
ไปทำงานที่ line number ตัวที่ n ใน list (n คือค่า integer ที่ได้จาก exp แล้วปัดเศษ)

OPEN mode, [#] file number, filename [,record length]
เปิดไฟล์เพื่อทำการ input หรือ output ตาม mode ที่กำหนด ถ้าไม่บอก record length จัดให้ 128 byte I = Sequential input, 0 = Sequential output, R = Random I/O

OPTION BASE n — กำหนดค่าต่ำสุดของ subscript ใน Array เป็น n (n = 0-1)

OUT port, byte (**ไม่มีใน Apple version)**
ส่วน byte ออกทาง output port ที่กำหนด (port และ byte = 0-255 อาจเป็น integer exp ก็ได้)

POKE address, byte — เก็บค่า byte ในหน่วยความจำ ณ แอดเดรสที่กำหนด (address = 0-65535, byte = 0-255 อาจเป็น integer exp ได้)

PRINT [USING"string format";] list of exp
พิมพ์ข้อมูลออกทางจอ ถ้าไม่มี USING และไม่มี list of exp จะได้บรรทัดว่าง

PRINT # file number, [USING "format";]list of exp
บันทึกข้อมูลลงใน seguential disk file string format ที่ใช้ใน USING มีดังนี้คือ

แทนตัวเลข *(Numeric field)*
. จุดทศนิยม
+ เครื่องหมายบวกหรือลบอยู่หน้าหรือต่อท้าย
— เครื่องหมายลบต่อท้าย (ถ้าเป็นลบ)
***asterisk** อยู่หน้าแทน **blank** ทุกตัว
$$ dollar sign 1 ตัวอยู่หน้าตัวเลข
***$$ asterisk** อยู่หน้าแทน **blank** ทุกตัว ยกเว้น 1 ตัวหน้าเลขเป็น **dollar sign**
, **comma** ทุก 3 ตัวหน้าจุดทศนิยม
↑↑↑↑ พิมพ์ค่าในรูป **exponent**
% แสดงว่าค่าที่พิมพ์เกิน **format** ที่กำหนด
! พิมพ์เฉพาะอักษรตัวแรกใน **string**
n space เว้นที่ว่า 2+n ตัวอักษรระหว่าง **string**
& พิมพ์ **string** ตามที่มีค่าอยู่ **(variable length)**

PUT [#] file number [,record number]
บันทึกข้อมูลจาก random file buffer ลงใน disk file ตาม record ที่กำหนด ถ้าไม่กำหนดก็เป็น record ถัดไป

RANDOMIZE [exp] จัดการสุ่มตัวเลขชุดใหม่โดยใช้ exp ที่ให้ค่าใหม่ ถ้าไม่มี exp จะถามค่าที่จะใช้ว่า Random Number Seed (range of address)?

READ list of variable อ่านข้อมูลจาก data statement และเก็บในตัวแปรตาม list

REM remark บอกว่าเป็นหมายเหตุ ไม่ใช่คำสั่งสำหรับ execute

RESTORE [line number] set pointer ที่ชี้ค่าใน DATA statement ให้กลับไปตั้งต้นใหม่เพื่ออ่านอีกเที่ยวหนึ่ง

RESUME หรือ **RESUME O**
กลับจาก error routine มาทำงานต่อที่ statement เดิมที่ error

RESUME NEXT กลับจาก error routine มาทำงานที่ line number ที่กำหนด

RETURN กลับจากโปรแกรมย่อยไปทำงานที่ statement ต่อจาก GOSUP

RSET string varible = string exp
จัด string เข้าใน buffer ของ random file โดยชิดขวา

STOP หยุดการทำงานตามโปรแกรม พิมพ์ line number ที่หยุด และกลับสู่ command level แต่ไม่ปิด file ต่าง ๆ

SWAP variable, variable สลับค่าระหว่าง variable สองตัวที่กำหนด ซึ่งต้องเป็น type เดียวกัน

WAIT port, mask (,Select)
ชะงักการทำงานตามโปรแกรม จนกว่าจะอ่านค่า input ทาง port ที่กำหนดแล้วมา X OR กับค่า select (default = 0) แล้วจึง AND กับ Mask ถ้าได้ค่าไม่เป็นศูนย์ จึงจะทำงานต่อไป
(ใน Apple Version จะเปลี่ยนจาก port เป็น address คืออ่านค่าจากแอดเดรสที่กำหนดแทนที่จะอ่านจาก port)

WHILE exp......WEND ทำงานตาม Statement ใน WHILE/WEND loop ตราบเท่าที่เงื่อนไขใน exp ยังเป็นจริง

WRITE (list of exp) พิมพ์ข้อมูลออกจากจอ

WRITE # File number, list of exp
บันทึกข้อมูลลงใน Sequential disk file

BASIC – 80 FUNCTION

ABS (exp) ให้ค่าสัมบูรณ์ของ exp

ASC (string) ให้ค่ารหัส ASCII ของตัวอักษรตัวแรกใน String

ATN (exp) ให้ค่า arctangent ของ exp ในหน่วยเรเดียน

CDBL(exp) เปลี่ยนค่า exp ให้เป็น double precision

CHR$ (exp) ให้ตัวอักษรที่ตรงกับ ASCII Code ที่กำหนดโดย exp

CINT (exp) เปลี่ยนค่า exp ให้เป็น integer

COS (exp) ให้ค่า Cosine ของ exp ในหน่วยเรเดียน

CSNG (exp) เปลี่ยนค่า exp ให้เป็น single precision

CVI (String) เปลี่ยนตัวอักษร 2 ตัว (2 byte) เป็น integer

CVS (string) เปลี่ยนตัวอักษร 4 ตัว (4 byte) เป็น single precision

CVD (string) เปลี่ยนตัวอักษร 8 ตัว (8 byte) เป็น Double precision

EOF (file number) ให้ค่า-1 เมื่อจบ file แล้ว

EXP (exp) ให้ค่า e ยกกำลัง exp

FIX (exp) ให้ค่า exp ที่ตัดเอาเฉพาะส่วนจำนวนเต็ม

FRE (O) ให้ค่าจำนวนหน่วยความจำที่ยังไม่ได้ถูกใช้โดย BASIC-80

FRE (String) ให้ค่าจำนวนหน่วยความจำที่เหลือในที่เก็บ string

HEX$ (exp) เปลี่ยนค่า exp ให้เป็นเลขฐานสิบหก

INKEY$ อ่านค่าตัวอักษร 1 ตัวจากที่คีย์บอร์ดถูกกด ถ้าไม่มีคีย์ใดถูกกดจะให้ null string

INP PORT อ่านค่าจาก input port ที่กำหนด (ไม่มีใน Apple Version)

INPUTS (length [,[#] File number)

อ่านค่า String ความยาว = length จากคีย์บอร์ด หรือ disk file (ถ้าระบุ file number)

INSTR ([position,] search, string, target string)

ให้ค่าตำแหน่งของ search string ที่ปรากฏใน target string โดยเริ่มค้นหาตั้งแต่ตำแหน่งที่กำหนดใน position (default เป็นตั้งแต่ต้น String)

INT (exp) ให้ค่าส่วนที่เป็นจำนวนเต็มของ exp

LEFT$ (String exp, length)

ให้ค่า string ความยาว = length ที่อยู่ซ้ายสุดในstring exp

LEN (string exp) ให้ค่าความยาวของ string

LOC (file number) ให้ค่า record ถัดไปที่จะ

อ่านหรือบันทึกใน random disk file หรือจำนวน Sector (128 byte block) ที่ใช้ไปแล้วใน sequential file

LOF (file number) ให้จำนวน Record ใน extent (128 record) สุดท้ายของ file นั้น

LOG (exp) ให้ค่า natural logarithm (ฐาน e) ของ exp

LPOS (exp) ให้ค่าตำแหน่งที่ printer จะพิมพ์เป็นอันดับต่อไป (exp เป็น dummy argument ไม่ใช้)

MID$ (string exp, starting position length)
ให้ string ที่เป็นส่วนของ string exp ตั้งแต่ starting position ไป ยาว = length

MKI $ (integer exp)

MKS$ (Single precision exp)

MKD$ (double precision exp)
เปลี่ยนค่าที่เป็นตัวเลขให้เป็น string (integer 2byte, single precision 4byte, double precision 8 byte)

OCT $ (exp) เปลี่ยนค่า exp ให้เป็นเลขฐาน 8

PEEK (address) ให้ค่าที่เก็บในหน่วยความจำ ณ แอดเดรสที่กำหนด

POS (exp) ให้ค่าตำแหน่งต่างของ cursor บนจอ ตำแหน่งซ้ายสุดเป็น 1

RIGHT$ (string exp, length)
ให้ค่า string ความยาว = length ที่อยู่ขวาสุดใน string exp

RND [(exp)] ให้ค่าเลขสุ่ม (random number)

SGN (exp) ให้ค่า 1 ถ้า exp>o,o ถ้า exp = o, -1 ถ้า exp<o

SIN (exp) ให้ค่า sine ของ exp ในหน่วยเรเดียน

SPACES (exp) ให้ string ที่ประกอบด้วยช่องว่าง (space) ยาว exp ตัว

SPC (exp) พิมพ์ blank จำนวน exp ตัว บนจอหรือ printer (ใช้กับ PRINT และ LPRINT)

SQR (exp) ให้ค่ารากที่สองของ exp

STR $ (exp) ให้ string ที่เขียนแทนค่าของ exp

STRING $ (length, string, exp)
ให้ string ที่ประกอบด้วยตัวอักษรที่มีรหัส ASCII = EXP หรือตัวอักษรตัวแรกใน string exp ต่อเนื่องกันยาว = length

TAB (exp) กำหนดตำแหน่งที่จะพิมพ์บนจอ และ printer (ใช้กับ PRINT และ LPRINT)

TAN (exp) ให้ค่า tangent ของ exp ในหน่วยเรเดียน

VSR[digit] (exp) เรียกใช้โปรแกรมย่อยภาษา Assembly ที่กำหนดในคำสั่ง DEF VSR (digit = 0-9 ถ้าไม่ใส่จะ default เป็น o)

VAL(string exp) ให้ค่าที่เป็นตัวเลขจาก string ที่เขียนแทนตัวเลขนั้น ๆ

VARPTR(variable) ให้แอดเดรสเริ่มต้นที่เก็บค่าตัวแปรที่กำหนด

VARPTR (#file number)
ให้แอดเดรสเริ่มต้นของ buffer สำหรับ file ที่กำหนด

Apple CP/M (SoftCard)version of BASIC-80

มี statement และ function เพิ่มเติมดังนี้คือ

STATEMENT

BEEP pitch, duration สร้างเสียงที่มีระดับซึ่งกำหนดโดย pitch และช่วงยาว = duration (pitch = 0 เสียงสูงสุด pitch = 255 เสียงต่ำสุด, duration = 0 สั้นที่สุด duration = 255 นานที่สุด)

CALL% variable [(argument list)]

เรียกโปรแกรมย่อยภาษา Assembly ของ CPU 6502 argument list มีได้ถึง 3 ตัว ส่งไปใน register A, X, และ y ตามลำดับ

COLOR = color number

เลือกสีที่จะใช้ในคำสั่งทางกราฟฟิค color number เป็นจำนวนเติมตั้งแต่ 0.15 มีสีดังนี้

0 black	8 brown
1 magenta	9 orange
2 dark blue	10 grey
3 purple	11 pink
4 dark green	12 green
5 grey	13 yellow
6 medium blue	14 agua
7 light blue	15 white

GET string variable อ่านค่าตัวอักษรตัวเดียวที่คีย์บอร์ดถูกกด แต่ไม่พิมพ์ออกมาบนจอ

GR screen number, Color number

HTAB screen position number

จัดจอสำหรับ low resolution graphic โดย clear จอเดิมทิ้ง screen number มีค่า 0 หรือ 1 (default เป็น 0) ถ้าเป็น 0 จะได้จอ 40 × 40 จุดและที่สำหรับข้อความ (text line) 4 บรรทัดล่าง ถ้าเป็น 1 จะได้ 40 × 48 จุด ไม่มี text line

HLIN x1, x2 **AT** Y ลากเส้นในแนวนอนจากจุด (x1, Y) ถึง (x2,Y)

HOME clear ข้อความทั้งหมดบนจอ cursor จะไปอยู่ที่มุมบนซ้าย

เลื่อน cursor ไปที่ screen ณ position ที่กำหนด

INVERSE เปลี่ยนลักษณะตัวหนังสือบนจอให้เป็นตัวคำบนพื้นฐาน

NORMAL เปลี่ยนลักษณะตัวหนังสือบนจอให้เป็นตัวขาวพื้นดำตามปกติ

PLOT X,Y plot จุด ณ ตำแหน่ง X,Y ที่กำหนด (0≥x≥39, 0≤y≤47)

POP RETURN ข้ามชั้นไปทำ statement หลังจากคำสั่ง CALL ครั้งก่อนครั้งสุดท้าย

TEXT เปลี่ยนจอจาก low หรือ High Rexolution Graphic กลับมาเป็นตัวหนังสือตามเดิม

VLIN Y1,Y2 ATx ลากเส้นในแนวดิ่งจาก (X,Y1) ถึง (X,Y2)

VTAB SCREEN line number

เลื่อน cursor ไปยังบรรทัด

ที่ต้องการบนจอ (กำหนดโดย Screen line number)

FUNCTION

BUTTON (I) ให้ค่าที่อ่านจากปุ่มบน game controller ตัวที่ I (I = 0-3) ถ้ากดได้ค่า -1 ไม่กดได้ 0

PDL (I) ให้ค่าที่อ่านจาก paddle ตัวที่ (I = 0-3) ค่าที่ได้อยู่ในช่วง 0-255

SCRN (X,Y) ให้ค่าตัวเลขที่บอกสี ณ จุด X,Y ที่กำหนดบนจอ ($0 \leq x \leq 39$, $0 \leq y \leq 47$)

VPOS (I) ให้ค่าตำแหน่งของ cursor ในแนวดิ่ง (1 บรรทัดบนสุด) I เป็น dummy argument

นอกจากนี้ใน GBASIC ซึ่งเป็น High Resolution Graphic version ของ Apple MBASIC ยังมี function และ statement เพิ่มอีกดังนี้คือ

HCOLOR = color number

เลือกสีที่จะใช้คำสั่ง High Resolution Graphic โดย Color number = 0-12 มีสีดังนี้

0 black	4 black	8 black 1
1 green	5 orange	9 white 1
2 violet	6 blue	10 black 2
3 white	7 white	11 white 2
		12 reverse

HGR screen number, Color number

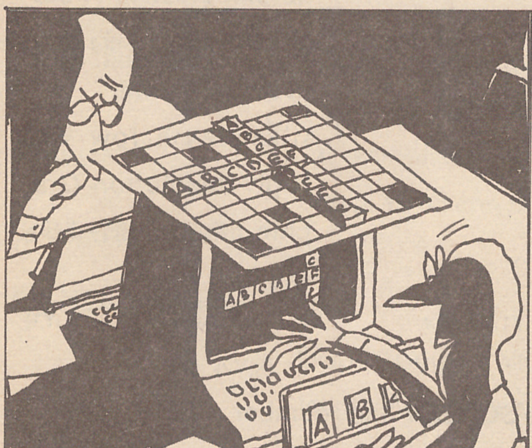
1. จะ clear screen และได้จอ 280 × 192 จุด ไม่มี text line
2. ไม่ clear screen และได้จอ 280 × 160 จุด + 4 text line
3. ไม่ clear screen และได้จอ 280 × 192 จุด ไม่มี text line

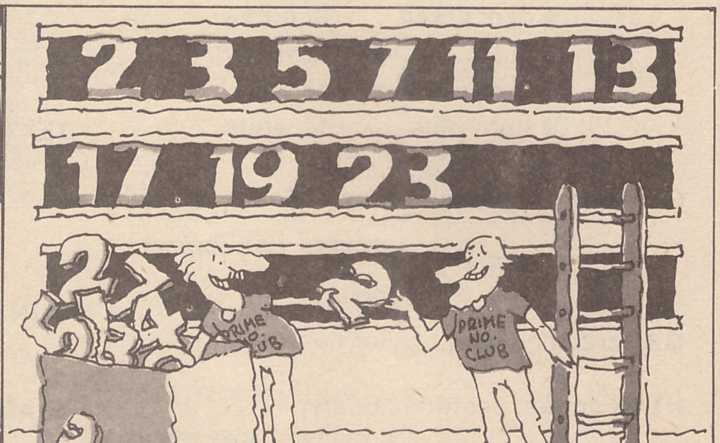
HPLOT X,Y plot จุด ณ ตำแหน่ง X,Y

HPLOT [X1,Y1] TO X2,Y2 [TO X3,Y3.....]

plot จุดเรียงต่อกันจาก x1,y1 (หรือจุดที่ plot ไว้ครั้งสุดท้าย ถ้าไม่กำหนด x1,y1) ไปยังจุด x2,y, และต่อ ๆ ไป

HSCRN (X,Y) ให้ค่า-1 ถ้าจุดx,y บนจอถูก plot ถ้าไม่เช่นนั้นให้ค่า 0 ∅





THE GAME OF LIFE

โดย สมคิด บุญประภากร

"LIFE" เป็นเกมส์ที่ออกแบบโดย J.H. Conway และเริ่มมีชื่อเสียงเป็นที่รู้จักกันทั่วไปเมื่อถูกตีพิมพ์ในคอลัมน์เกมส์ทางคณิตศาสตร์ในหนังสือ "Scientific American" เมื่อเดือนตุลาคม 1970 โดยเกมส์ LIFE ได้เสนอแนวความคิดของ time simulation ซึ่งเป็นหลักการเบื้องต้นในการจำลองการทำงานของตัวแบบ (MODEL) ต่าง ๆ ที่แปรผันโดยตรงกับเวลาด้วยคอมพิวเตอร์ โดยเมื่อ 10 กว่าปีก่อนนี้แนวความคิดแบบนี้ยังเป็นสิ่งใหม่และมีผู้ที่เข้าใจไม่มากนัก แต่ต่อมาเมื่อมีการพัฒนาของไมโครคอมพิวเตอร์อย่างขนานใหญ่เกิดขึ้นทำให้มีผู้นำแนวความคิดเรื่อง time simulation ไปใช้ในการพัฒนาซอฟท์แวร์ต่าง ๆ มากขึ้น ยกตัวอย่างเช่น Logic simulator (ระบบจำลองการทำงานของวงจรตรรก), Flight simulator (ระบบจำลองการบิน) และ package จำพวก CAD-CAM (Computer-Aided Design, Computer-Aided Manufacturing)

หลักการ

การเล่นเกมส์ LIFE จะกระทำบนตารางสี่เหลี่ยมที่แบ่งเป็นช่อง ๆ แต่ละช่องจะมีเซลของสิ่งมีชีวิตที่ยังมีชีวิตอยู่อยู่ภายในหรือไม่ก็เป็นช่องว่างเปล่า (ซึ่งหมายถึง เซลนั้นตายไปแล้ว) วิธีเล่นก็คือใส่เซลที่มีชีวิตเข้าไปในตารางที่ว่างเปล่าเมื่อเริ่มต้นการเล่น แล้วเฝ้าดูการเปลี่ยนแปลงของเซลในตารางนี้ในแต่ละช่วงเวลาที่ผ่านไป ตารางใหม่ที่เกิดขึ้นจะประกอบขึ้นจากการใส่สถานะใหม่ (เกิดใหม่หรือตายลง) ของแต่ละเซลในตารางโดยขึ้นกับสถานะเดิมของเซลนั้นและจำนวนของเซลที่มีอื่น ๆ ที่ล้อมรอบตัวมันอยู่ (แต่ละเซลจะมีเซลอื่น ๆ ล้อมรอบอยู่ได้ 8 ตัว คือเซลที่อยู่ข้างบน, ข้างล่าง, ด้านข้าง 2 ด้านและด้านมุมทั้ง 4 ของช่องนั้น) กฏเกณฑ์ที่จะกำหนดการเปลี่ยนแปลงสถานะ

```
10  PRINT CHR$(12)          'CLEAR SCREEN
15  PRINT CHR$(27)+"2"      'OFF CURSOR
20  PRINT " THE GAME OF LIFE "
30  PRINT " BY J.H. CONWAY "
40  DEFINT A-Z
50  SIZE=10
60  DIM GRID(11,11)
70  DIM OLD(11,11)
80  FOR I=0 TO SIZE+1
90      FOR J=0 TO SIZE+1
100         GRID(I,J)=0
110     NEXT J
120 NEXT I
130 PRINT "INPUT LIST OF CO-ORDINATES OF LIVE CELLS"
140 PRINT "THE GRID IS ";SIZE;" BY ";SIZE
150 PRINT "CO-ORDINATE PAIR CONTIANING A ZERO TERMINATES THE LIST."
160 INPUT "X,Y-ORDINATE ";X,Y
170 IF X=0 AND Y=0 THEN 200
180 GRID(X,Y)=1
190 GOTO 160
200 PRINT: INPUT "HOW MANY GENERATIONS";NUMBER
210 PRINT
220    FOR GEN=1 TO NUMBER
230        FOR X=0 TO SIZE+1
240            FOR Y=0 TO SIZE+1
250               OLD(X,Y)=GRID(X,Y)
260            NEXT Y
270        NEXT X
280        FOR X=1 TO SIZE
290            FOR Y=1 TO SIZE
300              NEIGHBOR=OLD(X-1,Y-1)+OLD(X-1,Y)+OLD(X-1,Y+1)
                      +OLD(X,Y-1)+OLD(X,Y+1)
                      +OLD(X+1,Y-1)+OLD(X+1,Y)+OLD(X+1,Y+1)
310              IF NEIGHBOR=3 OR NEIGHBOR=2 AND GRID(X,Y)=1 THEN GRID(X,Y)=1
                                                                 ELSE GRID(X,Y)=0
320            NEXT Y
330        NEXT X
340        PRINT CHR$(12)
350        PRINT "               ------------"
360        FOR ROW=1 TO SIZE
370           PRINT"              !";
380              FOR COLUMN=1 TO SIZE
390                 IF GRID(ROW,COLUMN)=0 THEN PRINT " ";
                                         ELSE PRINT "*";
400              NEXT COLUMN
410           PRINT "!"
420        NEXT ROW
430        PRINT "               ------------"
440    NEXT GEN
450 PRINT CHR$(27)+"3"      'ON CURSOR
460 END
```

ของแต่ละเซลเป็นดังนี้

●ถ้าเซลนั้นมีชีวิตอยู่และมีเซลอื่นล้อมรอบอยู่น้อยกว่า 2 เซล มันจะตายไปเพราะความโดดเดี่ยว

●ถ้าเซลนั้นมีชีวิตอยู่และมีเซลอื่น ๆ ล้อมรอบมากกว่า 3 เซลแล้วมันก็จะตายเพราะความแออัดและ.....

●ถ้าเซลนั้นตายไปแล้วและมีเซลที่มีชีวิตเซลอื่นล้อมรอบอยู่ 3 ตัวพอดี มันจะกลับมามีชีวิตใหม่อีก ขอยกตัวอย่างของตารางที่มีรูปร่างดังนี้

DEAD	DEAD	DEAD	DEAD
DEAD	LIVE	DEAD	LIVE
LIVE	LIVE	LIVE	DEAD
DEAD	DEAD	LIVE	DEAD

ตารางที่เกิดขึ้นถัดมาจะเป็นดังนี้

DEAD	DEAD	DEAD	DEAD
LIVE	LIVE	DEAD	DEAD
LIVE	DEAD	DEAD	LIVE
DEAD	DEAD	LIVE	DEAD

ขอให้คุณพิจารณาดูแต่ละเซล และทำความเข้าใจถึงเหตุผลของแต่ละเซลที่ตายไปหรือเกิดขึ้นใหม่

การหาตารางที่กำหนดขึ้นใหม่

หลักการที่ได้กล่าวมาจะใช้อัลกอริทึมดังต่อไปนี้

READ inintial configuration ; อ่านข้อมูลรูปร่างของตารางเริ่มแรก

READ number of generation, G ; รับข้อมูลจำนวนของตารางที่จะถูกสร้างขึ้นใหม่

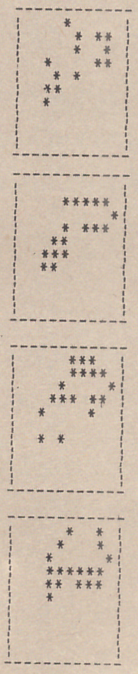
FOR each of G generation DO ; สำหรับทุก ๆ ตารางที่สร้างขึ้น

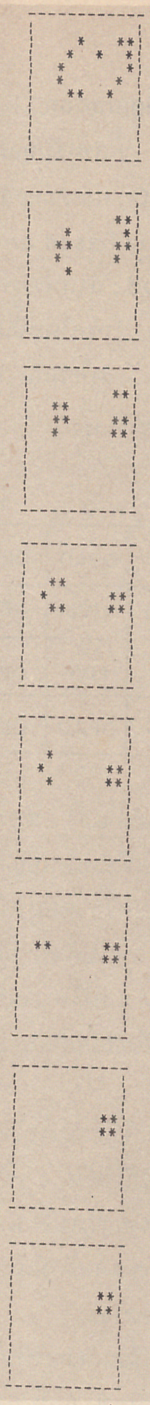
generate next configuration ; สร้างตารางที่มีรูปร่างใหม่ขึ้น

WRITE configuration ; พิมพ์ตารางออกมาบนจอภาพ

```
ABCDEFGHIJKLMNOP
QRSTUVWXYZ123


 THE GAME OF LIFE
 BY J.H. CONWAY
INPUT LIST OF CO-ORDINATES OF LIVE CELLS
THE GRID IS  10  BY  10
CO-ORDINATE PAIR CONTIANING A ZERO TERMINATES THE LIST.
X,Y-ORDINATE  2 , 4
X,Y-ORDINATE  4 , 2
X,Y-ORDINATE  8 , 4
X,Y-ORDINATE  4 , 8
X,Y-ORDINATE  6 , 4
X,Y-ORDINATE  4 , 6
X,Y-ORDINATE  2 , 6
X,Y-ORDINATE  5 , 5
X,Y-ORDINATE  2 , 8
X,Y-ORDINATE  8 , 2
X,Y-ORDINATE  3 , 7
X,Y-ORDINATE  4 , 8
X,Y-ORDINATE  1 , 9
X,Y-ORDINATE  8 , 4
X,Y-ORDINATE  3 , 6
X,Y-ORDINATE  3 , 7
X,Y-ORDINATE  3 , 9
X,Y-ORDINATE  5 , 8
X,Y-ORDINATE  5 , 2
X,Y-ORDINATE  5 , 4
X,Y-ORDINATE  5 , 6
X,Y-ORDINATE  1 , 4
X,Y-ORDINATE  0 , 0
HOW MANY GENERATIONS    20
```





เราจะต้องใช้ Matrix (ซึ่งก็คือ array ขนาด 2 มิติ) เพื่อแทนตารางของเซลเนื่องจากแต่ละเซลจะมีค่าเพียง 2 อย่าง (เป็นหรือตาย) เราจึงอาจใช้ Matrix ชนิด BOOLEAN ซึ่งใช้ค่า "FALSE" แทนเซลที่ตายแล้วและค่า "TRUE" แทนเซลที่ยังอยู่ **หรือเราอาจใช้ Matrix ชนิดที่เป็นจำนวนเต็มโดยให้ค่า 0 แทนเซลที่ตายและค่า 1 แทนเซลที่ยังอยู่ก็ได้**

สำหรับการสร้างตารางรูปใหม่ (generate next configuration) นั้นจะเป็นส่วนที่ยากที่สุดของอัลกอริทึมนี้ซึ่งจะได้อธิบายต่อไป ก่อนอื่นเราลองมาคิดวิธีที่จะสร้างตารางใหม่ขึ้นมาโดยคิดแบบตรง ๆ ตามอัลกอริทึมนี้

"generate next configuration"

FOR each now DO ; สำหรับทุกแถว

FOR each column DO ; และทุกคอลัมน์ในแถวนั้น

determine new value based on neighborhood ; กำหนดสถานะใหม่ของเซลโดยขึ้นกับเซลที่เป็นเพื่อนกัน (อยู่ล้อมรอบ)

แต่ขอให้พิจารณาผลลัพธ์ของตารางต่อไปนี้

แถว \ ช่องที่	1	2	3	4
1	LIVE	DEAD	DEAD	DEAD
2	DEAD	DEAD	LIVE	DEAD
3	LIVE	LIVE	LIVE	DEAD
4	DEAD	DEAD	DEAD	DEAD

สมมุติว่าเรากำลังคำนวณที่แถว 2 ช่องที่ 1 อยู่ ด้วยเหตุที่เซลนี้ตายไปแล้ว และมีเพื่อนบ้านอยู่ 3 เซลที่มีชีวิต ดังนั้นเซลนี้จึงกลับมามีชีวิตอีกครั้ง ต่อมาเมื่อคำนวณมาถึง แถวที่ 3 ช่องที่ 2 เซลนี้จะตายไปด้วยความแออัดเพราะมีเซลอื่นล้อมรอบอยู่ 4 ตัว ทั้ง ๆ ที่มันควรจะไม่ตายเพราะที่จริงแล้วในตารางก่อนหน้านี้มีเซลอื่นล้อมรอบตัวมันเพียง 3 ตัว

ด้วยเหตุนี้จึงแสดงว่าเราไม่สามารถใช้ Matrix ชุดเดียวสำหรับใช้เป็นตารางเพื่อแก้ไขค่าต่าง ๆ ในขณะที่กำลัง process ตารางนั้น เหตุผลก็คือการเปลี่ยนค่าเซล ๆ หนึ่งอาจมีผลกระทบไปยังเซลอื่น ๆ รอบตัวมันที่ถูกพิจารณาทีหลังในลำดับของการ execute ดังนั้นอัลกอริทึมพิมพ์จึงผิดพลาด เราจึงต้องเปลี่ยนอัลกอริทึมใหม่โดย copy ตารางไปไว้อีกทีหนึ่งเพื่อใช้เป็นข้อมูลในการสร้างตารางใหม่ถัดไป แล้วคำนวณค่าของเซลโดยขึ้นกับสถานะใน Matrix เก่าที่ถูก copy มาแล้ว

"generate next configuration"

Copy Geid To Old ; Copy ตารางไปยัง Old Matrix

FOR each row DO ; สำหรับทุกแถว

FOR each column DO และทุกคอลัมน์ในแถวนั้น

determine new value for grid based on neighborhood in Old

```
 THE GAME OF LIFE
 BY J.H. CONWAY
INPUT LIST OF CO-ORDINATES OF LIVE CELLS
THE GRID IS  10  BY  10
CO-ORDINATE PAIR CONTIANING A ZERO TERMINATES THE LIST.
X,Y-ORDINATE  5 , 2
X,Y-ORDINATE  3 , 6
X,Y-ORDINATE  2 , 5
X,Y-ORDINATE  9 , 9
X,Y-ORDINATE  8 , 8
X,Y-ORDINATE  7 , 7
X,Y-ORDINATE  6 , 6
X,Y-ORDINATE  5 , 5
X,Y-ORDINATE  4 , 4
X,Y-ORDINATE  3 , 3
X,Y-ORDINATE  2 , 2
X,Y-ORDINATE  1 , 1
X,Y-ORDINATE  10 , 5
X,Y-ORDINATE  9 , 6
X,Y-ORDINATE  8 , 4
X,Y-ORDINATE  3 , 6
X,Y-ORDINATE  1 , 4
X,Y-ORDINATE  8 , 2
X,Y-ORDINATE  2 , 2
X,Y-ORDINATE  1 , 5
X,Y-ORDINATE  4 , 8
X,Y-ORDINATE  9 , 2
X,Y-ORDINATE  7 , 5
X,Y-ORDINATE  7 , 6
X,Y-ORDINATE  7 , 4
X,Y-ORDINATE  9 , 2
X,Y-ORDINATE  5 , 2
X,Y-ORDINATE  5 , 4
X,Y-ORDINATE  6 , 3
X,Y-ORDINATE  8 , 3
X,Y-ORDINATE  2 , 8
X,Y-ORDINATE  2 , 4
X,Y-ORDINATE  2 , 7
X,Y-ORDINATE  0 , 0
HOW MANY GENERATIONS    25
```

```
------------
|  ****    |
| *    *   |
|  *  * *  |
|   *      |
|   **     |
|  *   *   |
| *   **   |
| * *   *  |
| * **     |
|          |
------------
```

```
------------
|  ****    |
| *    *   |
|  *   *   |
|  **      |
|  ***     |
|  *** *   |
| *   ***  |
|** * **   |
|   **     |
|          |
------------
```

```
------------
|  ****    |
| *  * *   |
| ***      |
| *  *     |
| *   *    |
| *    **  |
|**     *  |
|** *   *  |
|  ****    |
|          |
------------
```

```
------------
|  ****    |
| *        |
|** ***    |
|** **     |
|***  **   |
| **   **  |
|       ** |
|*  *  *   |
| ****     |
|   **     |
------------
```

กำหนดสถานะใหม่ของเซล
โดยขึ้นกับเซลที่เป็นเพื่อน
บ้านใน Old Matrix

ต่อไปเราจะคิดหาอัลกอริทึมในการคำนวณสถานะของแต่ละเซลในตาราง ซึ่งก็คือบรรทัดสุดท้ายของอับกอริทึมที่เพิ่มผ่านมานั่นเอง

ขอให้กลับไปศึกษากฎเกณฑ์ที่กล่าวไว้ตอนต้นอีกทีหนึ่ง จากกฎเหล่านี้เราสามารถสร้างตารางของสถานะใหม่ของเซลได้ดังนี้

	จำนวนเซลเพื่อนบ้าน	สถานะเดิม	
		DEAD	LIVE
สถานะใหม่	0	D	D
	1	D	D
	2	D	L
	3	L	L
	4	D	D
	5	D	D
	6	D	D
	7	D	D
	8	D	D

จากตารางนี้เราสามารถเขียนอัลกอริทึมได้ดังนี้

"determine new value for grid based on neighborhood in Old"

Compute number of neighbor, N

IF N = 3 or (N = 2 and cell was live ;

ถ้า N = 3 หรือ N = 2 และเซลมีชีวิต

THEN cell is live ; ดังนั้นเซลจะมีชีวิต

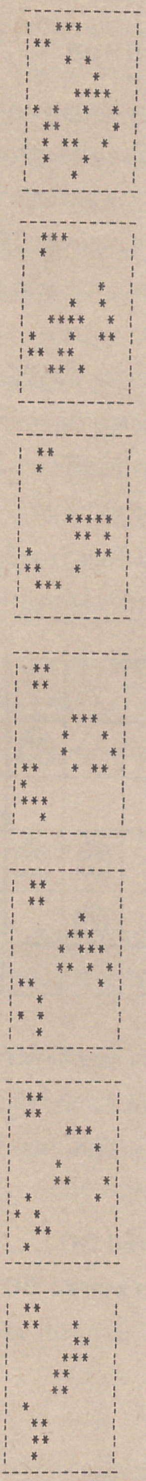
ELSE cell is dead ; ไม่เช่นนั้นเซลจะตาย

ส่วนที่เหลืออยู่ก็คือการคำนวณหาจำนวนเซลเพื่อนบ้านที่ล้อมรอบเซลนั้นอยู่ถ้าตารางของเซลเหล่านี้แทนด้วยตัวแปรจำนวนเต็ม เราจะเขียนอัลกอริทึมได้เป็น

"Compute number of neighbor, N"

Set N to sum of entries in 8 neighbor cells

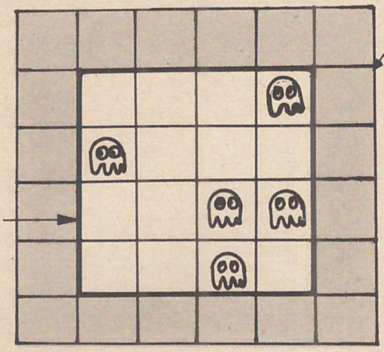
ถ้าหากว่าตารางของเซลเป็นตัวแปร BOOLEAN เราจะไม่สามารถทำการบวกมันได้และจะต้องมีอัลกอริทึมที่ซับซ้อนขึ้นไปอีก ดังนั้นเราจึงให้ตารางนี้เป็นตัวแปรจำนวน



เต็มเพื่อให้โปรแกรมง่ายขึ้น เมื่อเป็นเช่นนี้ เราจึงสามารถเขียนโปรแกรมที่บวกค่าในช่องรอบ ๆ เวลานั้นทั้ง 8 ช่องเข้าด้วยกันในบรรทัดเดียว

อัลกอริทึมของส่วนที่เหลือ

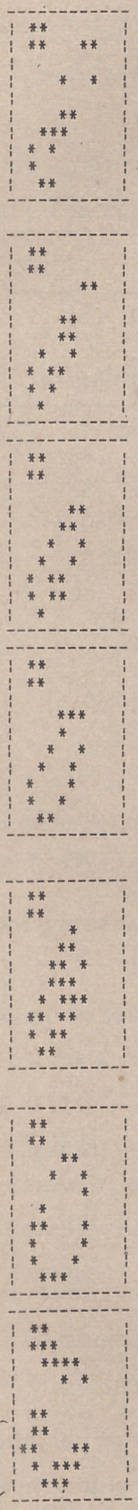
มีรายละเอียดบางอย่างที่คุณอาจสังเกตเห็นแล้ว คือ อัลกอริทึมที่เราสมมุติให้มีเซลที่ล้อมรอบแต่ละเซลอยู่ 8 เซล แล้วจะเกิดอะไรขึ้นกับเซลที่อยู่ตามขอบของตาราง ซึ่งมีเซลเพื่อนบ้านอยู่น้อยกว่า 8 เซล วิธีแก้ไขวิธีหนึ่งก็คือ เราอาจจะสร้างอัลกอริทึมเฉพาะแต่ละบางของเซล (คือ เซลที่อยู่ภายใน, ขอบซ้าย, ขอบขวา, ขอบบน, ขอบล่าง, และเซลที่มุมทั้ง 4 ของตาราง) แต่วิธีนี้ยุ่งยากซับซ้อนเกินไป วิธีอื่นที่ดีกว่านี้คือขยายตารางออกอีก 1 ช่องทุก ๆ ด้าน และให้ช่องที่ขยายออกเก็บเซลที่ตายแล้วเอาไว้ และเราจะไม่แก้ไขหรือพิมพ์เซลที่ขอบตาราง เพราะเซลเหล่านี้มีไว้เพียงเพื่อให้สามารถใช้อัลกอริทึมเดียวกันในทุก ๆ เซลได้ (ทุกเซลจะมีเซลเพื่อนบ้านล้อมรอบอยู่ 8 เซล) ดังรูป

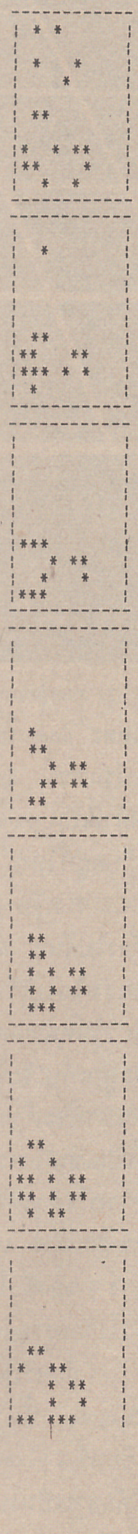


หลังจากที่เราได้สร้างอัลกอริทึมเพื่อคำนวณสถานะใหม่ของแต่ละเซลแล้วเราจะยังกลับไปดูส่วนที่ทำการ copy ตารางขึ้นมาอีกชุดหนึ่ง โดยเขียนอัลกอริทึมได้ดังนี้

```
"copy grid into Old"
  FOR each row stored DO ;
    FOR each column stored DO ;
      set Old value to grid value ; ใส่ค่าใน Old
                                    ตามค่าใน
                                    ตารางเดิม
```

ส่วนที่เหลือของโปรแกรมก็คือการอ่านข้อมูลที่ใช้สร้างตารางตอนเริ่มแรก (read initial configuration) โดยเราจะสร้าง loop ซ้อนกัน 2 loop สำหรับแต่ละแถวและแต่ละตารางเพื่อกำหนดค่าเริ่มต้นของสถานะของเซล





แต่ละเซลเป็น o (หมายถึงเซลที่ตายแล้ว) แล้วรับข้อมูลซึ่งเป็นพิกัดทางแกน X และ Y ของเซลที่ยังมีชีวิตอยู่ซึ่งเราใส่เข้าไป ดังนั้นอัลกอริทึมจึงเป็นดังนี้

```
"read initial configuration"
    FOR each row stored DO
        FOR each column stored DO
            set grid value to dead
READ I, J
WHILE I not O and J not O DO
    set grid value in row I, column J to live
    READ I, J
```

ส่วนสุดท้ายของโปรแกรมคือการพิมพ์ตารางออกมา (write configuration) โดยเราจะกำหนดให้ "*" แทนเซลที่ยังมีชีวิตในตาราง และ "   " (ช่องว่าง) แทนเซลที่ตายแล้ว เราสามารถเขียนอัลกอริทึมได้ดังนี้

```
"write configuration :"
    write upper bound of"-"s ; พิมพ์ขอบตาราง
                               ด้านบน
  FOR each row DO
    write row representation ; พิมพ์สัญญลักษณ์
                               บนแต่ละแถว
    write lower bound of "-" s ; พิมพ์ขอบตาราง
                                 ด้านล่าง
"write row representation :"
    write "/" for left border ;  พิมพ์ขอบซ้ายของ
                                 ตาราง
    ROR each column DO
        IF cell is dead THEN write bland
                  ELSE write "*"
    write "/" for right border ; พิมพ์ขอบขวาของ
                                 ตาราง
```

โปรแกรมภาษา BASIC ข้างท้ายนี้เป็นการนำเอาอัลกอริทึมทั้งหมดมารวมกันและดัดแปลงแก้ไขให้ใช้งานได้ *ขอให้สังเกตว่า OUTPUT ที่ได้จากโปรแกรมนี้มีรูปร่างแตกต่างกันไปในแต่ละช่วงเวลาที่ผ่านไป และในที่สุดก็จะเข้าสู่ภาวะอยู่ตัวหรือเสถียรคือจะไม่มีการเปลี่ยนรูปร่างต่อไปอีก* เราสามารถนำหลักการของ LIFE ไปใช้ในการสร้างรูปภาพแปลก ๆ โดยเปลี่ยนแปลงกฎเกณฑ์ที่กำหนดสถานะของแต่ละเซลให้ซับซ้อนมากขึ้นได้ ∅

ภาษา BASIC ที่ใช้กับ APPLE II

(ตอนที่ 2)



โดย วศิน

3. MICROSOFT BASIC COMPILER (BASCOM.COM)

ในตอนที่แล้ว เราได้พูดถึง APPLE-SOFT BASIC กับ MICROSOFT BASIC (ที่แบ่งเป็น MBASIC กับ GBASIC) ซึ่งทั้งหมดนั้นเป็นตัวแปลภาษา BASIC แบบ Interpreter ในตอนนี้เราจะมาดูถึงภาษา BASIC อีกแบบหนึ่ง ซึ่งพัฒนาขึ้นโดยบริษัท MICROSOFT อีกเช่นกัน แต่เป็นตัวแปลภาษาแบบ Compiler ซึ่งมีประสิทธิภาพในการทำงานและมีความเร็วสูงกว่าตัวแปลภาษาแบบ interpreter แต่ก่อนอื่นขอทบทวนความเข้าใจของคุณผู้อ่านถึงเรื่องข้อแตกต่างระหว่างตัวแปลภาษาแบบ interpreter กับ Compiler สักเล็กน้อย ในฉบับที่ 3 เราเคยลงเรื่องของตัวแปลภาษาคอมพิวเตอร์ ทั้งแบบ Interpreter และ compiler มาแล้ว)

Interpreter VS.Compiler

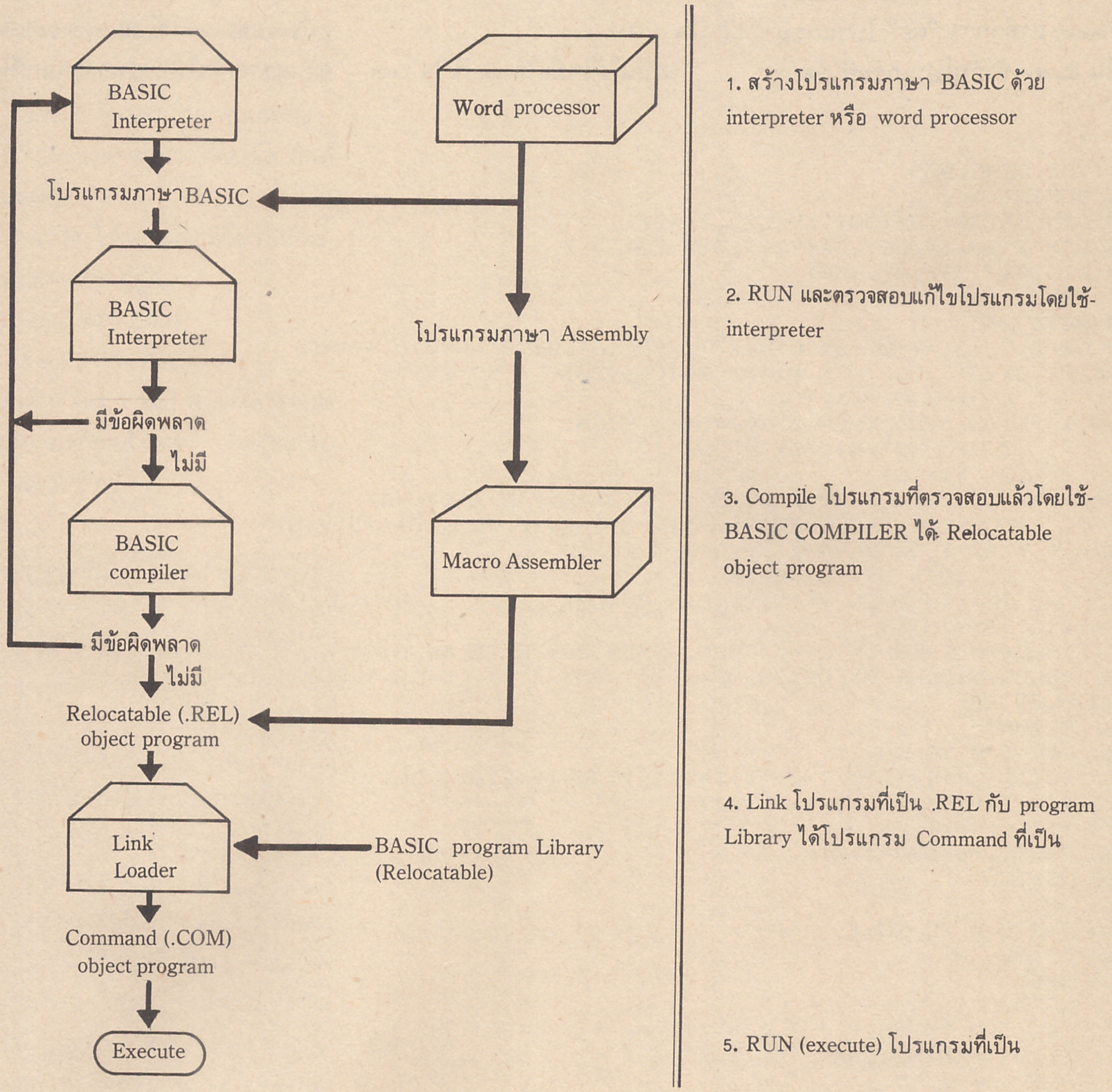
สำหรับตัวแปลภาษา BASIC ที่เป็น interpreter มันจะทำงานเมื่อคุณ RUN โปรแกรม (Runtime) โดยตัว interpreter จะดูว่าคำสั่งแต่ละ Statement คืออะไร และค้นหาโปรแกรม หรือส่วนของโปรแกรมที่ตรง กับคำสั่งนั้น (ซึ่งเก็บอยู่ในส่วนของตัว interpreter เอง) จากนั้นจึงทำงานตามโปรแกรมส่วนนั้น ๆ ถ้ามีตัวแปรหรือพารามิเตอร์ต่าง ๆ ที่เกี่ยวข้องกับการทำงาน ตัว interpreter ก็จะต้องไปค้นหาค่าของพารามิเตอร์เหล่านั้นในหน่วยความจำตามที่จัดเก็บเอาไว้ ขณะเดียวกันก็จะต้องตรวจสอบความถูกต้องของโปรแกรมไปด้วยจากนั้นจึงจะทำงานตามคำสั่งแต่ละคำสั่งได้

จะเห็นได้ว่า กระบวนการดังกล่าวต้องดำเนินไปจนครบรอบในการทำงานทุก ๆ คำสั่ง ถ้าคำสั่งที่ใช้มีการซ้ำกันเช่นใน loop ของ FOR/NEXT กระบวนการดังกล่าวก็จะถูกทำซ้ำ ๆ กันในแต่ละเที่ยวที่มีการทำคำสั่งนั้น นอกจากนี้การเก็บตัวโปรแกรมในลักษณะการทำงานแบบ interpreter จะเก็บเป็น linked list คือแต่ละ lines โยงกันเป็นทอด ๆ ที่ที่เก็บโปรแกรม line หนึ่งก็จะมีข้อมูลที่ทำหน้าที่เป็น pointer ชี้บอกว่า line ต่อไปถูกเก็บอยู่ที่ไหน เวลาทำงานตามคำสั่ง-GOSUB หรือ GOTO ตัว interpreter จะต้องไปค้นหา line ที่กำหนดตั้งแต่ต้นโปรแกรมไล่ไปเรื่อย ๆ จนกว่าจะเจอ ตัวแปรในหน่วยความจำก็เช่นเดียวกัน มันจะถูกเก็บเรียงกันไปเรื่อย ๆ และการค้นหาก็จะต้องไล่ตั้งแต่ต้นเช่นเดียวกัน ที่เป็นเช่นนี้เพราะตัว interpreter ไม่สามารถใส่ตำแหน่งที่แท้จริงในหน่วยความจำ (Absolute Address) ที่ตัวแปรหรือ line ต่าง ๆ ของตัวโปรแกรมเก็บอยู่ลงไปในตัวโปรแกรมได้

สำหรับตัวแปลภาษา BASIC แบบ Compiler นั้น จะแปลโปรแกรมที่คุณเขียนขึ้น (source program) จาก file หนึ่งให้กลายเป็นโปรแกรมภาษาเครื่อง (object program) ต่างหากอีก file หนึ่ง การแปลและตรวจสอบความถูกต้องทั้งหมด

จะกระทำในขั้นของการแปล (Compile-time)

file ใหม่ที่ได้นี้จะเป็นภาษาเครื่องซึ่งสามารถจะให้ CPU ทำงานได้ แต่ตำแหน่งของ line และตัวแปรต่าง ๆ ในหน่วยความจำยังไม่ได้ถูกกำหนดตายตัว ทั้งนี้เพราะตัวโปรแกรมนี้ยังไม่สมบูรณ์ เนื่องจากฟังก์ชันการทำงานบางอย่างของโปรแกรมจะเป็นภาษาเครื่องที่ยาวมาก และมีการใช้ซ้ำ ๆ กันในหลายส่วนของโปรแกรมผู้ผลิต Compiler จึงมักนิยมเขียนส่วนของโปรแกรมที่ทำฟังชันเหล่านั้นไว้ต่างหาก เก็บไว้รวม ๆ กันเป็น "ห้องสมุดโปรแกรม" (program library) เวลา compiler แปลโปรแกรมของคุณที่ต้องการทำฟังชั่นเหล่านี้ มันก็จะแปลเป็นเพียงคำสั่งเรียก (call) โปรแกรมย่อยใน library แทน แต่ในขณะนั้นโปรแกรมย่อยเหล่านั้นยังไม่ได้ถูก load เข้ามา จึงไม่อาจบอกได้ว่าเมื่อ load เข้ามาแล้วจะอยู่ที่แอดเดรสเท่าไร ดังนั้น ad-



รูปที่ 1 ขั้นตอนในการพัฒนาโปรแกรมภาษา BASIC

dress ต่าง ๆ ในคำสั่ง Call เหล่านี้จึงยังกำหนดลงไปไม่ได้ และโปรแกรมในลักษณะที่ยังไม่ได้กำหนดแอดเดรสแน่นอนนี้จึงถูกเรียกว่าเป็นโปรแกรมที่ย้ายที่ได้(Relocatable) หรือในระบบ CP/M มักกำหนดชนิดของ file เป็น .REL

หลังจาก Compiler ทำการแปลโปรแกรมเป็น file แบบ Relocatable แล้วจะต้องใช้โปรแกรมอีกอันหนึ่งที่เรียกว่า Link loader มาทำการ "ต่อ" โปรแกรมย่อยใน library เข้ากับโปรแกรมที่ได้จากการแปลครั้งแรก ทำให้ได้โปรแกรมที่ใหญ่ขึ้นและครบถ้วนสมบูรณ์ ตัว Link loader นี้จะ load เอาโปรแกรมจาก library เข้ามาต่อกับโปรแกรมจากการแปลครั้งแรก และคำนวณแอดเดรสที่ยังขาดอยู่ใส่ลงไปในโปรแกรม ดังนั้นโปรแกรมในขั้นนี้จึงถูกกำหนดแอดเดรสแน่นอนและพร้อมจะทำงานได้ทันที ซึ่งเรียกว่าเป็น Command file หรือในระบบ CP/M ย่อเป็น COM

จะเห็นได้ว่าเมื่อได้ file ที่เป็น Command file แล้ว โปรแกรมในลักษณะนี้จะทำงานได้เร็วกว่า interpreter แต่ต้องเสียเวลาในการแปล (compile) และการเชื่อมต่อโปรแกรม (link) ส่วนการทำงานของ interpreter จะใช้ RUN โปรแกรมได้ในทันที ดังนั้นในการใช้งานแล้ว จึงควรใช้ตัว interpreter ทดสอบแก้ไขโปรแกรมก่อนว่าทำงานได้ถูกต้องเหมาะสมหรือยัง ถ้าถูกต้องเหมาะสมแล้วจึงค่อย compile และ link ให้เป็น commond file เพื่อความรวดเร็วในการทำงานอีกทีหนึ่ง

```
A>TYPE DEMO.BAS
10 DEFINT A-Z
20 DIM SX(30),SY(30);XV(30),YV(30)
30 DIM X(30),Y(30),DXV(30),DYV(30)
40 DIM AXV(30),AYV(30)
50 NO=20
60 GOTO 200
70 OX=X(O):OY=Y(O):SX(O)=AXV(O)+SX(O):SY(O)=AYV(O)+SY(O)
100 IF SX(O)>256 THEN SX(O)=SX(O)-256:
      X(O)=X(O)+DXV(O)
110 IF SY(O)>256 THEN SY(O)=SY(O)-256:
      Y(O)=Y(O)+DYV(O)
120 X=X(O):Y=Y(O)
130 IF X>39 THEN X=78-X:XV(O)=-XV(O):DXV(O)=-DXV(O)
      ELSE IF X<0 THEN X=-X:XV(O)=-XV(O):DXV(O)=-DXV(O)
140 IF Y>47 THEN Y=94-Y:YV(O)=-YV(O):
      DYV(O)=-DYV(O) ELSE
      IF Y<0 THEN Y=-Y:YV(O)=-YV(O):          DYV(O)=-DYV(O)
170 X(O)=X:Y(O)=Y
174 IF OX=0 THEN IF OY=0 THEN GOSUB 300 ELSE IF OX=X THEN IF OY=Y THEN RETURN
190 COLOR=0:PLOT OX,OY:COLOR=(O+1) AND 15:IF O=15 THEN COLOR=1
195 PLOT X,Y
196 RETURN
200 FOR I=0 TO NO-1
210 XV(I)=20+I*(210/(NO+1)):YV(I)=230-I*(210/(NO+1))
212 AXV(I)=XV(I):AYV(I)=YV(I)
215 X(I)=1:Y(I)=1
216 DXV(I)=1:DYV(I)=1
220 NEXT
230 GR 1
240 FOR O=0 TO NO-1
250 GOSUB 70
260 NEXT
270 GOTO 240
300 FOR I=0 TO NO-1:J=J OR X(I) OR Y(I):NEXT
310 IF J THEN RETURN
320 BEEP 100,255
330 GOTO 330
```

รูปที่ 2 BASIC source program

มุม Apple II

ขั้นตอนในการพัฒนาโปรแกรมภาษา BASIC ที่สมบูรณ์เป็นดังนี้ภาพ

ขอให้สังเกตว่า เราอาจนำโปรแกรมภาษา Assembly มาใช้ร่วมกับโปรแกรมภาษา BASIC ที่ถูก COMPILE โดยใช้ตัวแปลภาษา Assembly ที่เรียกว่า Macro Assembler จากนั้นนำโปรแกรมที่ผ่านการแปลแล้วมา Link เข้าด้วยกัน

BASIC Compiler ของไมโครซอฟท์นี้ ถ้าจะพูดไปแล้วก็คือ Compiler ที่ใช้แปลโปรแกรมที่เขียนขึ้นด้วย MBASIC และ GBASIC นั่นเอง ในแพ็คเกจของ-BASIC Compiler จะประกอบด้วยดิสค์สามแผ่นคือ

1. BASCOM ใช้แผ่นนี้จะมีโปรแกรม-BASIC Compiler อยู่
2. MBASIC RUNTIME
3. GBASIC RUNTIME

ในแผ่นที่ 2 และ 3 จะมี Runtime Module (จะอธิบายภายหลัง)ของ MBASIC และ GBASIC อยู่

BASCOM

ในแผ่นนี้จะประกอบด้วย file ต่าง ๆ ดังนี้คือ

BASCOM.COM คือโปรแกรม BASIC Compiler (ขนาด 32 K)

L80. COM เป็นโปรแกรมที่จะทำหน้าที่ link และ load โปรแกรมที่ได้จากการ Compile (ขนาด 11 K)

OBSLIB.REL เป็น program library ภาษา BASIC ที่จะถูกดึงบางส่วนไป link กับโปรแกรมจากการ compile ดังนั้นส่วนต่าง ๆ ของ program library นี้จึงยังไม่ได้ถูกกำหนดaddress แน่นอน คือเป็น Relocatable (ขนาด 51 K)

M80. COM คือโปรแกรม Macro-

```
BASCOM 5.3Ø
Copyright 1981 (C) by MICROSOFT
.9194 Bytes Free
 ØØ14 ØØØ7        1Ø DEFINT A-Z
 ØØ14 ØØØ7        2Ø DIM SX(3Ø),SY(3Ø),XV(3Ø),YV(3Ø)
 ØØ14 ØØØ7        3Ø DIM X(3Ø),Y(3Ø),DXV(3Ø),DYV(3Ø)
 ØØ14 ØØØ7        4Ø DIM AXV(3Ø),AYV(3Ø)
 ØØ14 ØØØ7        5Ø NO=2Ø
       ** ØØ14'IØØØØØ: CALL     $53Ø
       ** ØØ17'LØØØ1Ø: LØØØ2Ø: LØØØ3Ø: LØØØ4Ø: LØØØ5Ø:
       ** ØØ17'        LD       HL,ØØ14
       ** ØØ1A'        LD       (NO%),HL
 ØØ1D Ø275        6Ø GOTO 2ØØ
       ** ØØ1D'LØØØ6Ø: JP       LØØ2ØØ
 ØØ2Ø Ø275        7Ø OX=X(O).OY=Y(O):SX(O)=AXV(O)+SX(O):SY(O)=AYV(O)+SY(O)
       ** ØØ2Ø'LØØØ7Ø: LD       HL,(O%)
       ** ØØ23'        ADD      HL,HL
       ** ØØ24'        PUSH     HL
       ** ØØ25'        LD       DE,X%
       ** ØØ28'        ADD      HL,DE
       ** ØØ29'        LD       E,(HL)
       ** ØØ2A'        INC      HL
       ** ØØ2B'        LD       D,(HL)
       ** ØØ2C'        EX       DE,HL
       ** ØØ2D'        LD       (OX%),HL
       ** ØØ3Ø'        POP      HL
       ** ØØ31'        PUSH     HL
       ** ØØ32'        LD       DE,Y%
       ** ØØ35'        ADD      HL,DE
       ** ØØ36'        LD       E,(HL)
       ** ØØ37'        INC      HL
       ** ØØ38'        LD       D,(HL)
       ** ØØ39'        EX       DE,HL
       ** ØØ3A'        LD       (OY%),HL
       ** ØØ3D'        POP      HL
       ** ØØ3E'        PUSH     HL
       ** ØØ3F'        LD       DE,AXV%
       ** ØØ42'        ADD      HL,DE
       ** ØØ43'        LD       E,(HL)
       ** ØØ44'        INC      HL
       ** ØØ45'        LD       D,(HL)
       ** ØØ46'        EX       DE,HL
       ** ØØ47'        LD       (T:Ø1),HL
       ** ØØ4A'        POP      HL
       ** ØØ4B'        PUSH     HL
       ** ØØ4C'        LD       DE,SX%
       ** ØØ4F'        ADD      HL,DE
       ** ØØ5Ø'        PUSH     HL
       ** ØØ51'        LD       E,(HL)
       ** ØØ52'        INC      HL
       ** ØØ53'        LD       D,(HL)
       ** ØØ54'        LD       HL,(T:Ø1)
       ** ØØ57'        ADD      HL,DE
       ** ØØ58'        EX       DE,HL
       ** ØØ59'        POP      HL
       ** ØØ5A'        LD       (HL),E
       ** ØØ5B'        INC      HL
       ** ØØ5C'        LD       (HL),D
       ** ØØ5D'        POP      HL
       ** ØØ5E'        PUSH     HL
       ** ØØ5F'        LD       DE,AYV%
       ** ØØ62'        ADD      HL,DE
       ** ØØ63'        LD       E,(HL)
       ** ØØ64'        INC      HL
       ** ØØ65'        LD       D,(HL)
       ** ØØ66'        EX       DE,HL
       ** ØØ67'        LD       (T:Ø2),HL
       ** ØØ6A'        POP      HL
       ** ØØ6B'        LD       DE,SY%
       ** ØØ6E'        ADD      HL,DE
       ** ØØ6F'        PUSH     HL
       ** ØØ7Ø'        LD       E,(HL)
       ** ØØ71'        INC      HL
       ** ØØ72'        LD       D,(HL)
       ** ØØ73'        LD       HL,(T:Ø2)
       ** ØØ76'        ADD      HL,DE
       ** ØØ77'        EX       DE,HL
       ** ØØ78'        POP      HL
       ** ØØ79'        LD       (HL),E
       ** ØØ7A'        INC      HL
       ** ØØ7B'        LD       (HL),D
 ØØ7C Ø27B        1ØØ IF SX(O)>256 THEN SX(O)=SX(O)-256:
 ØØ7C Ø27B            X(O)=X(O)+DXV(O)
       ** ØØ7C'LØØ1ØØ: LD       HL,(O%)
       ** ØØ7F'        ADD      HL,HL
```

รูปที่ 3 โปรแกรมเดียวกับในรูปที่ 2 เมื่อผ่านการ compile แล้วในภาพเป็นภาษา Assembly ของ Z-80 ซึ่งเทียบเท่ากับ object code ที่เป็นภาษาเครื่องใน Relocatable file และเป็น listing ที่ได้จาก listing file

Assembler ที่แปลโปรแกรมที่เขียนเป็นภาษา Assembly และภาษา Macro (ภาษาที่ใช้ร่วมกับ Assembly เพื่อรวมการทำงานตามโปรแกรมย่อยเข้าเป็นคำสั่งใหม่) ให้เป็น Relocatable object file (ขนาด 19 K)

CREF.COM โปรแกรมที่ใช้สร้าง CROSS REFERENCE LISTING จาก CROSS REFERENCE FILE ที่สร้างโดย-Macro Assembler (M80.COM)

Listing นี้จะมีรายละเอียดบางอย่างเพิ่มเติมเพื่อช่วยในการ debug โปรแกรม (ขนาด 4 K)

LIB80.COM โปรแกรมที่ช่วยจัดระบบของ program library โดยผู้เขียนโปรแกรมอาจสร้าง Library ของตนเองขึ้นใหม่ เพื่อเพิ่มความสะดวกในการ-link ก็ได้

ในการใช้งาน ขั้นแรกก็เป็นการเรียก BASCOM มา compile โปรแกรมภาษา BASIC โปรแกรมที่จะถูก compile ต้องอยู่ในรูป ASCII Code (ถ้าเขียนด้วย-interpreter เวลา SAVE ก็ใช้ ,A ต่อท้ายชื่อfile) ส่วน format ในการใช้งานเป็นดังนี้คือ

BASCOM ชื่อ Relocatable object file, ชื่อ listing file = ชื่อ BASIC source file เช่น-

BASCOM DEMO,DEMO = DEMO

BASCOM จะจัดการ compile โปรแกรมภาษา BASIC จาก file DEMO.BAS และได้ Relocatable object file ชื่อ DEMO.REL ขณะเดียวกันก็จะสร้าง listing file คือผลลัพธ์จากการ compile ที่อยู่ในรูปภาษา Assembly พร้อมทั้ง Address ของแต่ละคำสั่งในภาษา Assembly ด้วยในชื่อDEMO.PRN

โปรดสังเกตว่า BASCOM ตัวที่ใช้กับ Apple นี้เป็น version 5.30 และจะทำการ compile ทั้งหมดเป็นภาษา Assembly ของ Z-80 ไม่ใช่ 8080 ทั้งนี้เพราะ Soft Card ที่จะใช้กับโปรแกรมนี้เป็น Z-80 card เท่านั้น

ถ้าคุณต้องการกำหนด file extension(ตัวอักษร 3 ตัวหลังจุด)ในชื่อ file ต่าง ๆ ก็ให้กำหนดลงไปในตอนเรียก - BASCOM เลย ถ้าไม่เช่นนั้น BASCOM จะถือเอาว่า Source file ที่จะ compile จะมี file extension เป็น .BAS listing file เป็น .PRN และ Relocatable object file เป็น .REL

ในการ compile นี้มีสวิทช์หรือคำสั่งพิเศษที่จะใช้กับ compiler อีกหลายตัว โดยใช้/ต่อท้ายคำสั่ง และตามด้วยตัวอักษรต่าง ๆ เช่น BASCOM DEMO,-DEMO = DEMO/4/T สวิทช์เหล่านี้แบ่งได้เป็น 3 กลุ่ม มีหน้าที่การทำงานดังนี้คือ

ประเภท	สวิทช์	การทำงาน
1. Convention (เลือก convention ต่าง ๆ ในการ Compile)	/4	- ใช้การ Compile โดยยึดลักษณะการใช้คำ(lexical convention) Microsoft BASIC version 4.51 ซึ่งต่างจาก version 5.0 ที่ใช้อยู่เป็นปกติดังนี้คือ 1. ไม่จำเป็นต้องเว้นวรรคระหว่างคำที่เป็นคำสั่ง (keyword) 2. ชื่อตัวแปรต้องไม่มีส่วนใดซ้ำกับคำที่สงวนไว้ใช้เป็นคำสั่ง (reserve word) 3. ชื่อตัวแปรตั้งยาวได้ แต่มีความสำคัญเพียง 2 ตัวเช่น A11 กับ A 12 ถือเป็นตัวเดียวกัน การใช้ lexical convention ของ version 4.51 มีประโยชน์ในกรณีที่ source program เขียนติดกันหมด เช่น FORI = ATOBSTEPC ถ้าไม่ใช้ /4 compiler จะแปลเป็นตัวแปร ชื่อ FORI กับตัวแปรชื่อ ATOBSTEPC ถ้าใช้ /4 จึงจะแปลเป็น state

	/T	ment FOR - บอก compiler ให้แปลโปรแกรมโดยจัดคำสั่งต่าง ๆ ให้เกิดการทำงานตามลักษณะการทำงาน (excution convention) ของ version 4.51 ซึ่งต่างจาก version 5.0 คือ 1. FOR/NEXT loop จะถูก execute อย่างน้อยหนึ่งครั้ง (คือครั้งแรกโดยไม่เช็คค่า index ก่อน) 2. คำสั่ง TAB,SPC,POS,LPOS จะทำงานตามแบบของ version 4.51 3. การแปลงเลข floating point เป็นเลขจำนวนเต็ม (integer) จะใช้การตัด (truncation) แทนที่จะเป็นการปัด (rounding) 4. คำสั่ง INPUT จะไม่เปลี่ยนค่าในตัวแปรเดิมถ้ามีการตอบรับเพียงแต่กด Return เท่านั้นไม่ได้ป้อนค่าใหม่เข้าไป
	/C	- บอก compiler ให้ละการตรวจสอบ line number ดังนั้นใน source program อาจไม่ใส่ line number เลยก็ได้ โปรแกรมจะถูกทำไปตามลำดับคำสั่งตั้งแต่ต้น นอกจาก line ที่ถูกอ้างถึงในคำสั่ง GOTO หรือ GOSUB เท่านั้นที่ยังต้องคง line number ไว้ การลด line number ที่ไม่จำเป็นทำให้โปรแกรมอ่านง่ายขึ้น อนึ่งสวิทซ์ /4จะใช้ร่วมกับ /C ไม่ได้
2. Error trapping (จัดการกับ Error)	/E	- ในกรณีที่โปรแกรมของคุณมีคำสั่ง ON ERR GOTO อยู่จะต้องใช้ /E เพื่อบอก compiler ให้รู้และสร้าง line number address table เพื่อเก็บข้อมูลว่า line ใดของโปรแกรมอยู่ที่แอดเดรสใด เวลาเกิด error ในขณะ RUNจะได้รายงานได้ว่า Error ที่ line ใดซึ่งจะช่วยในการตรวจแก้ไข (debug) โปรแกรมได้มาก นอกจากนี้มีการใช้ ON ERR GOTO การแปลคำสั่ง GOSUB และ RETURN ทุกตัวจะต้องมี Code พิเศษเพิ่มเติมลงไปทั้ง Table และ Code เหล่านี้ทำให้โปรแกรมมีขนาดใหญ่ขึ้นอีก - ในกรณีที่มีคำสั่ง ON ERR GOTO เพื่อไปทำโปรแกรมย่อยที่จะจัดการกับ error แล้วก็มักจะมีคำสั่งให้กลับไปทำงานในโปรแกรมต่อ คือ RESUME ซึ่งถ้าใช้ RESUME line number คือสั่งให้กลับไปทำคำสั่งใน line ที่ระบุไว้ก็ใช้ /E ได้ แต่ถ้ามี RESUME, RESUME O (สั่งให้กลับไปทำ line ที่เกิด error ใหม่) หรือ RESUME NEXT (กลับไปทำ line ถัดจากที่เกิด error) จะต้องใช้

		/X แทน /E
3. Special Code	/N	- ทำให้ listing file ไม่มีส่วนที่เป็นโปรแกรมภาษา Assembly ของ Z -80 คงมีแต่เพียงโปรแกรมภาษา BASIC ที่เป็น soure และแอดเดรส (สัมพัทธ์กับจุดเริ่มต้นโปรแกรม คือนับว่าโปรแกรมเริ่มที่แอดเดรส 0)ทำให้ listing file มีขนาดเล็กลง
	/0	- ใช้บอก compiler ว่า program library ที่จะ link เข้ากับ Relocatable object code จากการ compile นี้จะเป็น file ชื่อ OBSLIB.REL ซึ่งจะทำให้ได้โปรแกรมใน command file ที่สมบูรณ์ไม่ต้องมีโปรแกรมสนับสนุนใด ๆ อีก ซึ่งทำให้ไม่สามารถใช้ Runtime module ได้ (ถ้าใช้ Runtime Module แล้วการ link จะใช้ program library อีกตัวหนึ่งชื่อ BASLIB.REL ขนาด 32 K
	/D	- ทำให้โปรแกรมที่ compile สามารถรายงานความผิดพลาดที่เกิดขึ้นในขณะ RUN ได้ และสามารถใช้คำสั่ง TRACE/NOTRACE ในโปรแกรม เพื่อช่วยในการ debug ได้ ถ้าไม่ใช้ /D compiler จะไม่สนใจคำสั่งดังกล่าว นอกจากนี้ /D ยังทำให้ compiler แปลคำสั่งในการทำงานต่าง ๆ ในลักษณะที่ยาวกว่าและทำงานช้ากว่าเดิมเพื่อให้สามารถตรวจสอบข้อผิดพลาดต่อไปนี้ 1. Arithmatic overflow ค่าที่คำนวณได้มากหรือน้อยเกินกว่าที่จะเก็บได้อย่างถูกต้อง ทั้ง integer และ floating point 2. Array bound ใช้ subscript ของตัวแปรเกินกว่าขนาด array ที่ DIMension ไว้ 3. line number ถูกรวมเข้าไว้ใน object file เพื่อให้รายงานได้ว่าเกิดที่ line ใด 4. คำสั่ง RETURN จะถูกตรวจว่ามี GOSUB ที่คู่กันหรือไม่ ถ้าไม่ใช้ /D แล้ว การใช้ subscipt เกินขนาด Array, การ overflow จากการคำนวณ ตลอดจนการ RETURN โดยไม่มี GOSUB จะไม่ทำให้มีการรายงานความผิดพลาดใด ๆ ทั้งในการ Compile และ RUN แต่ผลลัพธ์ที่ได้จะผิดไป
	/S	- ถ้าโปรแกรมของคุณมีข้อความ (string) ในเครื่องหมายคำพูดอยู่เป็นจำนวนมาก compiler จะเก็บ string เหล่านี้ไว้ใน Memory ก่อนยังไม่ write ลงบนดิสค์ ถ้ามี string เข้ามาซ้ำกับของเดิม

compiler จะจัดการเก็บเพียงตัวเดียว และ write ลงบนดิสค์ทีหลัง ซึ่งวิธีการนี้ถ้า string มีจำนวนมากหรือยาวอาจทำให้ memory ไม่พอ การใช้ /S จะทำให้ compiler จัดการ write string ที่พบลงบนดิสค์ทันทีที่พบวิธีนี้ทำให้ใช้ที่ใน memory น้อยลงขณะ compile แต่ถ้า string มีการซ้ำกันจะถูกเก็บไว้ทุกตัว ทำให้ file ที่ได้จากการ compile มีขนาดใหญ่ขึ้น เวลา RUN ก็ต้องการที่ในหน่วยความจำมากขึ้น

การ Link

เมื่อ compile เสร็จต่อไปก็เป็นขั้นตอนของการ link โดยใช้ L80 ดังนี้

L80 ชื่อ Relocatable object file [,ชื่อ-Command file] [switch] [switch].....เช่น L80 DEMO,DEMO /N/E

ซึ่งจะทำให้ L80 ทำการ load โปรแกรม DEMO.REL และบางส่วนของ program library ที่เป็น .REL เข้ามาจัดการ link จากนั้น /N จะทำให้ L 80 จัดการ save file ที่ได้เป็น command file เพียง file เดียว ซึ่งอยู่ในสภาพพร้อมที่จะ execute ได้โดยการเรียกชื่อ file เท่านั้น ส่วน /E จะทำให้การทำงานกลับสู่ CP/M อีกครั้งหนึ่ง นอกจากนี้ยังมี /G ซึ่งเป็นสวิทซ์ที่ทำให้โปรแกรมนั้นถูก execute ทันทีต่อจากการ link เมื่อเสร็จแล้วจึงกลับสู่ CP/M นอกจากนี้ ในกรณีที่คุณไม่ต้องการเก็บ command file ก็ใช้เพียง /G และไม่ต้องใส่, ชื่อ command file และ /N ลงไป เช่น

L80 DEMO/G

L80 จะทำการ load และ link เมื่อเสร็จแล้วก็ execute ทันทีโดยไม่ save command file ให้ ดังนั้นถ้าคุณต้องการ RUN โปรแกรมนี้ใหม่ก็ต้องเรียก L 80 มา link ใหม่อีกครั้งหนึ่ง

นอกจากนี้แล้ว L 80 ยังมีสวิทซ์ต่าง ๆ อีกหลายอย่าง ซึ่งมีลักษณะการใช้งานต่างออกไป ทั้งนี้เพราะ L 80 นี้นอกจากจะใช้กับ BASCOM แล้วยังสามารถใช้ link โปรแกรมในรูป Relocatable ที่ได้จากการ compile โปรแกรมภาษา FORTRAN, COBOL ตลอดจน Assembly ของไมโครซอฟท์ได้อีกด้วย การใช้งาน-L 80 ที่ยกมาเป็นเพียงส่วนหนึ่งที่ใช้มากในการใช้งานกับ BASCOM เท่านั้น สวิทซ์ต่าง ๆ ของ L 80 จะขออธิบายแต่เพียงคร่าว ๆ ดังนี้คือ

หน้าที่	สวิทซ์	การทำงาน
GO	/G /G:filename	- ทำการ Execute โปรแกรมที่ได้จากการ link เมื่อเสร็จก็กลับสู่ CP/M - ทำการ Execute เหมือน /G โดยถือเอา file ที่ระบุชื่อไว้นั้นเป็น file แรกหรือ Main Program ที่จะทำ Execute (ในกรณีที่มี .REL มา link กันหลาย ๆ file)
Exit	/E /E:filename	- กลับสู่ CP/M - กลับสู่ CP/M เหมือน /E แต่จะ set แอดเดรสตั้งต้นของ file ที่ระบุชื่อไว้นั้นให้เป็นจุดเริ่มต้นการ Execute ใน command file (ในกรณีที่มี .REL มา link กันหลาย ๆ file)

หน้าที่	สวิทช์	การทำงาน
Save	/N	- Save โปรแกรมที่ได้จากการ link ทั้งหมดลงใน disk เป็น command file เพียง file เดียว
	/N:P	- ตามปกติ file จะประกอบด้วย 2 ส่วนคือส่วนที่เป็นโปรแกรม กับส่วนที่เป็นที่เก็บข้อมูล /N:P จะทำให้มีการ save เฉพาะส่วนที่เป็นโปรแกรม (ตัวชุดคำสั่ง) เท่านั้น ไม่ save ส่วนข้อมูล ทำให้ file มีขนาดเล็กลง
Address setting	/p:address	- กำหนดแอดเดรสตั้งต้นของ memory ที่จะ save โปรแกรมลงไป ถ้าใช้คู่กับ /D จะกำหนดเฉพาะแอดเดรสที่จะ save ส่วนของโปรแกรม(program area) เท่านั้นไม่รวมส่วนเก็บข้อมูล (data area)
	/D:address	- กำหนดแอดเดรสตั้งต้นของ data area
	/R	- ทำการ Reset L80 ให้กลับไปคอยรับคำสั่งใหม่อีกรอบหนึ่ง
Global listing	/U	- undefine globals หรือสัญญลักษณ์ซึ่งถูกอ้างถึงทั่วไปในโปรแกรมอื่น (ในกรณีที่มี .REL หลาย file มา link กัน) และยังไม่ได้ถูกกำหนดค่าแน่นอน จะถูก list ออกมา
	/M	- จัดการ list เอา global ทั้งหมด ทั้งที่ define และ undefine ออกมา
library Search	/S	- ค้นหาส่วนของโปรแกรมเพื่อทำการ link จาก library ใน file ที่มีชื่ออยู่นำหน้า /s
Radix Setting	/O	- บอก L80 ว่าตัวเลขที่ใช้ในการป้อนข้อมูลให้ทั้งหมดเป็นฐาน 8 (octal)
	/H	- ตามปกติแล้วตัวเลขที่ใช้จะเป็นฐาน 16 (Hexadecimal)
Special Code	/X	-Save .Com file ในรูปแบบรหัส ASC II ฐานสิบหกของ Intel โดยกำหนดให้ file ที่ save มี extensive เป็น .HEX
	/Y	- สร้าง file เพื่อการ debug ด้วย SID/ZSID debugger โดยมี extenstion เป็น .SYM

ถึงตอนนี้คุณคงจะได้ .COM file ที่สามารถจะทำงานได้รวดเร็วกว่าเดิมแล้ว อย่างไรก็ตามภาษา BASIC ที่ใช้กับ-Compiler ก็ยังมีข้อแตกต่างจากที่ใช้กับ interpreter อยู่ ซึ่งจะได้กล่าวถึงหลังจากการอธิบายเกี่ยวกับการใช้ Runtime Module ต่อไปส่วนการใช้งาน M 80.COM, CREF.COM และ LIB80.COM นั้นจะเกี่ยวข้องกับโปรแกรมภาษา Assembly ค่อนข้างมาก และไม่ค่อยจะได้ใช้ในการพัฒนาโปรแกรม BASIC ทั่ว ๆ ไป จึงขอ

ละไว้ไม่นำมาอธิบายในบทความนี้ หากคุณต้องการค้นคว้าเพิ่มเติมจะหาได้จากUtility Software manual ของไมโครซอฟท์

Runtime Module

ถ้าคุณลอง compile และ linkโปรแกรมภาษา BASIC ตามที่อธิบายมาแล้ว จะพบปัญหาประการหนึ่งว่าการ compile นั้นใช้เวลาไม่มากนัก แต่การ link ใช้เวลานานทั้งนี้เพราะใน library นั้นจะประกอบด้วยโปรแกรมย่อยที่เรียกว่า module เพื่อทำหน้าที่ตามคำสั่งต่าง ๆ โปรแกรมที่ผ่านการ compile จะเรียกใช้โปรแกรมย่อยหรือ module เหล่านั้น และในแต่ละ-module อาจเรียกใช้ module อื่น ๆ อีก เช่นคำสั่ง PRINT SIN (1.5) ในโปรแกรมจากการ compile จะเรียกใช้ module ที่ทำคำสั่ง PRINT และใน module นี้ก็จะต้องเรียก module ที่ทำฟังก์ชัน SIN อีกทีหนึ่ง ตัว link loader จะต้องไปค้นหา module เหล่านี้ใน library จัดการ load เข้ามา และใส่แอดเดรสตามตำแหน่งที่อยู่จริงใน-memory ให้แก่คำสั่ง call ที่เรียกใช้กันระหว่าง module ต่าง ๆ การค้นหาใน library นี้อาจต้องค้นกลับไปกลับมาหลายเที่ยว



เช่น module ที่ call อยู่ตอนกลางของ library แต่ module ที่ถูก call อยู่ตอนต้น ตัว link loader ก็จะต้องกลับมาหา module ตอนต้นใหม่ การทำงานของ link loader นี้จะช้ามาก และนับเป็นปัญหาที่สำคัญประการหนึ่งในการทำงานของไมโครคอมพิวเตอร์

เพื่อลดเวลาที่ต้องใช้ในการ link ลงจึงได้มีการพัฒนา Runtime Module ขึ้น แนวความคิดของมัน ก็คือ ถ้า module ที่ถูกใช้งานย่อย ๆ จะอยู่ในรูปที่กำหนดแอดเดรสแน่นอนเลย (คือไม่ใช่ Relocatable แต่เป็นแบบ command file) การทำงานของ link loader ก็เพียงแต่เอาแอดเดรสที่แน่นอนเหล่านั้นไปใส่ไว้ในคำสั่ง call ในโปรแกรมจากการ compile ถ้า module ที่ถูกกำหนดแอดเดรสแน่นอนนี้เรียกใช้กันเอง ตัว link loader ก็ไม่ต้องไปเกี่ยวด้วย ส่วนถ้ามีการเรียกใช้ module อื่น ๆ ที่ไม่ได้กำหนดแอดเดรสแน่นอน (Relocatable) ตัว link loader ก็จึงค่อยไปจัดการ-load เข้ามาและ link ตามวิธีเดิม ดังนั้นโอกาสที่ link loader จะต้องจัดการ link กับ module ที่ไม่ได้กำหนดแอดเดรสแน่นอนจึงลดลงได้มาก ทำให้การทำงานเร็วขึ้น แต่เนื่องจากมีการกำหนดแอดเดรสแน่นอนลงไป Module เหล่านั้นจึงต้องถูก load เข้าไป ใน memory ตามแอดเดรสที่กำหนดไว้ในขณะ RUN จะ load ไปที่อื่นไม่ได้

ในทางปฏิบัติ module ที่ใช้บ่อย ๆ ได้ถูกรวมเป็น 1 file เป็น .COM เรียกว่า MRUN.COM สำหรับ MBASIC (ขนาด 16K) และ GRUN.COM สำหรับ GBASIC (ขนาด 17 K กับเนื้อที่สำหรับเก็บภาพบนจอ อีก 8 K รวม เป็น 25 K) ส่วน module ที่ใช้ไม่บ่อยนักก็ถูกรวมเป็น library อันใหม่ชื่อ BASLIB.REL (ขนาด 26 K สำหรับ MBASIC และ 27 K สำหรับ GBASIC) ดังนั้น disk แผ่นที่เป็น Runtime Module ของ MBASIC และ GBASIC จะประกอบด้วย file ต่าง ๆ ดังนี้คือ

-MRUN.COM(16K) หรือ GRUN.COM (17K) คือตัว Runtime module สำหรับส่วนของโปรแกรมที่ใช้บ่อย ๆ

- BASLIB.REL program library ภาษา BASIC สำหรับส่วนของโปรแกรมที่ถูกใช้ไม่บ่อยนัก BASLIB ใน disk ของ MBASIC (26 K) เป็นคนละตัวกับ BASLIB ใน disk ของ GBASIC (27 K)

- BCLOAD (1 K) เก็บข้อมูลแอดเดรสที่ใช้ในการ load โปรแกรม และ Runtime Module (ของ MBASIC และ GBASIC จะไม่เหมือนกัน)

- DEMO.BAS (2 K) โปรแกรมตัวอย่าง ภาษา BASIC เก็บในรูป ASCII file

-L80.COM (11 K) คือ link loader ตัวเดียวกับในแผ่น BASCOM

การ compile โปรแกรมตามปกติจะใช้ Runtime module อยู่แล้ว โดย L 80 จะไปค้นหา module อื่น ๆ ใน BASLIB.REL แต่ถ้าไม่ต้องการใช้ Runtime module ก็ใช้/0 ในตอน compile เมื่อมีการ link L80 จะไปหา module ทั้งหมดที่ต้องใช้ใน OBSLIB.REL

เมื่อมีการใช้ Runtime module การจัด memory จะเป็นดังรูปที่ 4

ข้อดีในการใช้ Runtime Module คือ

1. คำสั่ง CHAIN จะทำงานคงตามความหมายคือโปรแกรมเดิมกับโปรแกรมที่ถูก CHAIN จะถูก Execute ต่อเนื่องกันจริง ๆ

2. คำสั่ง COMMON จะสามารถใช้เชื่อมโยงการทำงานระหว่างโปร

แอดเดรสสุดท้ายของ memory

CP/M
Memory **ที่เหลือ**
ส่วนที่เก็บข้อมูลซึ่งใช้โดย Runtime module
โปรแกรมของผู้ใช้
ข้อมูลในโปรแกรมของผู้ใช้
เก็บตัวแปรที่ใช้ในคำสั่ง COMMON
RUNTIME MODULE MRUN.COM(16K) **หรือ** GRUN.COM (17K+8K **สำหรับจอภาพ**)
CP/M vector

แอดเดรสตั้งต้นใน memory

รูปที่ 4 การจัด memory เมื่อใช้ Runtime module

แกรมที่มีการ CHAIN กันได้เพราะมีเนื้อที่สำหรับตัวแปรในคำสั่ง COMMON จัดไว้ต่างหาก

3. ลดเวลาที่ใช้ในการ link ลง

4. Runtime module ถูกกำหนดแอดเดรสแน่นอนแล้ว จึงไม่จำเป็นต้องถูก load เข้ามาใน memory ในขณะ link ทำให้มีเนื้อที่ใน memory มากขึ้นและ COM file ของคุณที่ได้จากการ link มีขนาดเล็กลงเพราะไม่รวม Runtime module แต่ในขณะ RUN คุณจะต้องมี Runtime module อยู่บนดิสค์เพื่อให้พร้อมที่จะ load เข้าไปหลังจาก .COM file ถูก load แล้ว

ส่วนข้อดีในการใช้ OBSLIB.REL คือไม่ใช้ Runtime module ก็มีดังนี้คือ

1. โปรแกรมมีความสมบูรณ์ในตัว เวลา RUN ไม่จำเป็นต้อง load Runtime module จาก disk ในขณะ RUN

2. สามารถใช้ /P และ /D กำหนดตำแหน่งตั้งต้นสำหรับเก็บโปรแกรมและ data ในหน่วยความจำได้ จึงสามารถนำโปรแกรมไปใช้กับระบบที่ใช้ ROM โดยเก็บส่วนของโปรแกรมใน ROM (เพราะไม่ต้องมีการแก้ไข) และส่วนของ data ใน RAM (เพราะต้องแก้ไขเปลี่ยนแปลงตามการทำงานของโปรแกรม)

3. สำหรับโปรแกรมเล็ก ๆ เมื่อ link แล้วอาจมีขนาดเล็กกว่า Runtime module จึงเหมาะกับการเก็บใน ROM มากกว่าและข้อเสียในการใช้ OBSLIB.REL ก็มีดังนี้คือ

1. COMMON ใช้ตัวแปรร่วมกันระหว่างโปรแกรมที่มีการ CHAIN ไม่ได้

2. CHAIN เป็นการสั่งให้ RUN โปรแกรมอีกโปรแกรมหนึ่งเท่านั้น ไม่ได้ execute ต่อเนื่องกันจริง ๆ

3. ไม่มีคำสั่ง CLEAR,RUN<line number>

ความแตกต่างในการใช้งาน BASIC interpreter กับ compiler

จากที่ผ่านมาเราได้กล่าวถึงองค์ประกอบและการใช้งาน BASIC compiler ของไมโครซอฟท์ (BASCOM) ไปแล้ว ในตอนนี้จะขอพูดถึงความแตกต่างระหว่างภาษา BASIC ของไมโครซอฟท์ที่ใช้กับ interpreter และ compiler ซึ่งแบ่งได้เป็น 3 ประเภทใหญ่ ๆ ดังนี้คือ

1. ความแตกต่างในการทำงาน

เนื่องจากการใช้งานโปรแกรมภาษา-BASIC ที่ถูก compile แล้วนั้น กระทำโดยการเรียกใช้ผ่าน CP/M เลย(โดยพิมพ์ชื่อโปรแกรมที่เป็น .COM)และไม่เกี่ยวข้องกับ source file ที่เป็นภาษา BASIC ดังนั้นคำสั่งที่เป็น direct mode เกี่ยวข้องกับ file และการแก้ไขโปรแกรม (editieng) จึงไม่สามารถทำงานได้คำสั่งเหล่านี้คือ

AUTO	CONT	DELETE	EDIT
ERASE	LIST	LLIST	LOAD
MERGE	NEW	RENUM	SAVE

2. ความแตกต่างที่ตัวภาษา

การทำงานของโปรแกรมตามคำสั่งภาษา BASIC ต่อไปนี้จะแตกต่างกันระหว่าง interpreter กับ compiler คือ

1. CALL ในภาษา BASIC ของไมโครซอฟท์ ปกติจะเป็นการเรียกใช้โปรแกรมย่อยภาษา Assembly ของ Z-80 เท่านั้น แต่ในการใช้งานกับ compiler นั้น-CALL ยังอาจใช้เรียกโปรแกรมย่อยที่เป็นภาษา FORTRAN ได้อีกด้วย คือภาษา FORTRAN-80 ซึ่งเป็นแพ็คเกจอีกวันหนึ่งของไมโครซอฟท์เช่นกัน โดยชื่อโปรแกรมย่อยที่เรียกจะต้องปรากฏในประโยค FORTRAN SUBROUTINE ในโปรแกรมย่อย หรือเป็นสัญลักษณ์ที่กำหนดให้เป็น PUBLIC Symbol ในภาษา-Assembly ที่ใช้กับ MACRO-80 ของไมโครซอฟท์จึงจะเรียกใช้ได้โดย L80 link loader

จะสามารถจัดการ load และ link ให้

ส่วนการเรียกใช้โปรแกรมย่อยภาษา Assembly ของ CPU 6502 ของ-Apple โดยใช้คำสั่ง

CALL%<variable name>[(<argument>)]

นั้นยังคงใช้ตามเดิม

2. CHAIN ไม่สามารถใช้ option ต่าง ๆ คือ MERGE,ALL,DELETE<range> และ line number ได้อย่างใน interpreter ทั้งนี้เพราะจะเป็นการ CHAIN ไปยังโปรแกรมที่เป็นภาษา BASIC ที่ Compile แล้วเหมือนกันจึงไม่สามารถอ้างอิงถึงตัวแปร (variable) และ line number ต่าง ๆ ได้ การส่งผ่านค่าของตัวแปรไปยังโปรแกรมที่ถูก CHAIN จึงต้องใช้คำสั่ง-COMMON แทน และ file ต่าง ๆ จะถูกเปิดทิ้งไว้ไม่มีการปิด file ในการ CHAIN

3. CLEAR ใช้ได้เฉพาะเมื่อใช้กับ-Runtime Module เท่านั้นและ expression ที่กำหนดตำแหน่งสูงสุดที่จะใช้ได้ใน Memory และ <expression2> ที่กำหนดขนาดสูงสุดของ stack ซึ่งเป็น option ให้เลือกใช้จะต้องเป็น integer expression ถ้า<expression 1>ให้ค่า 0 จะถือว่าเป็นตำแหน่งสูงสุดใน memory ที่เคยกำหนดไว้เดิม และถ้า <expression 2> เป็น 0 จะกำหนดให้ stack มีขนาด 256 ไบท์

คำสั่ง CLEAR จะปิดไฟล์ทั้งหมด set ค่าในตัวแปรที่เก็บตัวเลขทั้งหมดให้เป็นศูนย์ และตัวแปรเก็บตัวตัวอักษรเป็นnull string (ความยาวเป็น 0) รวมทั้งที่อยู่ในประโยค COMMON ด้วย

4. COMMON เป็นคำสั่งที่ให้ compiler จัดที่สำหรับตัวแปรในระหว่างการ-compiler เท่านั้นไม่ได้ถูกแปลเป็นคำสั่งให้ทำงานใด ๆ ในระหว่างการ execute (เป็น non-executable statement) ประโยคที่ execute ไม่ได้เหล่านี้ก็มี

```
COMMON
DEFDBL,DEFINT,DEFSNG,
DEFSTR
DIM
OPTION BASE
REM
% INCLUDE
```

ในการใช้ COMMON กับ Array ประโยค COMMON ต้องมาก่อน DIM ที่กำหนด Dimension ของ ARRAY และการแบ่งตัวแปรที่ COMMON ไว้เป็นกลุ่มย่อย ๆ โดยตั้งชื่อ ตามแบบภาษา FORTRAN ก็อาจทำได้ดังรูปแบบต่อไปนี้

COMMON /<ชื่อ>/<ตัวแปร 1, ตัวแปร 2,...>

ซึ่งจะใช้ได้ในการติดต่อกับโปรแกรมย่อยภาษา FORTRAN และ Assembly แต่ไม่สามารถใช้กับการ CHAIN ได้ ชื่อที่ใช้จะต้องยาวไม่เกิน 6 ตัว และเริ่มด้วยตัวอักษร ส่วนถ้าไม่มีการตั้งชื่อจะถือว่าเป็นช่องว่าง (blank)

5. DEF type (DEFine INTeger/SiNGle precision/DouBLe precision/STRing)

ในการทำงานของ compiler จะเป็นการจัดที่ในหน่วยความจำให้กับตัวแปรต่าง ๆ ที่ปรากฏเป็นพารามิเตอร์ในคำสั่งเหล่านี้ในช่วงที่คำสั่งเหล่านี้มีผล แต่จะไม่แปลออกมาเป็นคำสั่งใด ๆ ในภาษาเครื่องเลย ดังนั้นคำสั่งเหล่านี้ผลตอน compile เท่านั้น เมื่อกำหนด Type แล้วจะมีผลไปตลอดจนกว่าจะพบคำสั่ง DEFtype ใหม่ และเหตุที่ DEFtype ไม่มีการ execute ในตอน RUN การใช้คำสั่ง GOTO ในโปรแกรมเพื่อข้ามคำสั่ง DEF type ไปจึงไม่มีผลทำให้ข้ามการกำหนด-type และจัดที่ใน memory ให้ตัวแปรซึ่งถูกทำเรียบร้อยตั้งแต่ตอน compile แล้วไปได้ ส่วนตัวแปรที่กำหนด type แน่นอนด้วยเครื่องหมาย %, !, #, $ ต่อท้ายชื่อจะไม่ได้รับผลใด ๆ จากคำสั่งนี้เช่นเดียวกับใน interpreter

6. DIM (DIMension)

เป็น non-executable statement ที่มีผลให้ compiler จัดที่ให้แก่ตัวแปรในตอน-compiler เท่านั้นโดยปกติจะมีค่า default คือ set ไว้เป็น 10 ถ้ามีการใช้ Array โดยไม่ได้ DIMeusion ไว้และถ้ามีการ DIMension ซ้ำจะเกิด error ที่บอกว่า "Array Already Dimensioned" ขึ้น

DIM จะต้องอยู่ก่อนหน้าคำสั่งที่มีการใช้ Array นั้น ๆ จะแยกเป็น subprogram เอาไปไว้ตอนท้ายโปรแกรมก็ไม่ได้ เพราะถ้าเป็นเช่นนั้น compiler จะอ่านพบการใช้ Array ก่อนการ DIMension และกำหนดขนาดของ Array ตามที่ default ไว้เป็น 10 แทนที่จะเป็นตามที่เราต้องการ

นอกจากนี้พารามิเตอร์ในประโยค DIM จะต้องเป็นค่าคงที่ที่เป็นจำนวนเต็ม (numeric constant) เท่านั้นจะเป็นตัวแปรหรือนิพจน์ (expression) ใด ๆ ไม่ได้ เพราะจะต้องกำหนดที่แน่นอนลงไปตอน compile และ Array ที่ได้จะมีขนาดคงที่

7. END เป็นคำสั่งที่ใช้จบการทำงานตามโปรแกรม โดยปิดไฟล์ทั้งหมด และกลับไปยังโปรแกรมควบคุมระบบ CP/M ถ้าในโปรแกรมไม่มีคำสั่ง END ตอนท้าย compiler จะจัดการใส่การทำงานตามคำสั่ง END ให้เองในโปรแกรมที่ compile ได้

8. ERASE เป็นคำสั่งที่ใช้ลบการ-DIMension ที่ทำไว้และทำให้สามารถกำหนดขนาดใหม่ (redefine)ของ Array ได้ ซึ่งเป็นสิ่งที่ทำในตอน compiler เท่านั้น

```
FOR I     FOR J
 |          |
 |          |       FOR K
 |        NEXT J      |
 |                    |
 |                  NEXT K
NEXT I
```

```
FOR I=1 TO 10
INPUT A
IF A*I < 10 THEN NEXT I
B = A/I
NEXT I
```

ไม่มีในการ RUN จึงใช้ compiler ไม่ได้

9. FOR/NEXT loop

สามารถใช้ Double precision loop control ได้ และ FOR/NEXT loop จะต้องถูกกำหนดซ้อนกันอย่างคงที่ (staticcally nested) นั่นคือเมื่อมีประโยค FOR หนึ่งแห่งจะต้องมี NEXT เพียง 1 แห่งเท่านั้นที่คู่กับ FOR นั้นและ FOR/NEXT คู่หนึ่งจะต้องอยู่ภายใน loop ของคู่อื่น จะซ้อนคาบเกี่ยวกันไม่ได้ ดังเช่นโปรแกรมตัวอย่างจะใช้ไม่ได้

10. % INCLUDE เป็นคำสั่งเฉพาะของ Compiler มีรูปแบบคือ

% INCLUDE <ชื่อ file>

คำสั่งนี้จะทำให้ compiler ไปทำการรวมเอา Source file อื่นในภาษา-BASIC มาร่วมในการ compile ด้วย ชื่อ file ที่ใช้จะต้องเป็นภาษา BASIC เท่านั้น และไม่ต้องใส่เครื่องหมายคำพูดในชื่อ file ส่วน File extension หลังจุดจะเป็น .BAS

โปรแกรมที่จะถูก INCLUDE เข้ามา compile ร่วมกับโปรแกรมที่มีคำสั่ง-% INCLUDE นั้นจะต้องไม่มีคำสั่ง END อยู่และ line number ต้องใช้กันได้กับโปรแกรมที่สั่ง INCLUDE เช่นไม่ซ้ำหรืออยู่ในช่วงเหลื่อมกันนอกจากนี้การใช้ INCLUDE มีข้อกำหนดดังนี้คือ

- file ของโปรแกรมที่ถูก INCLUDE จะต้อง SAVE เป็น ASCII code
- line number ในโปรแกรมที่ถูก-INCLUDE แต่ละโปรแกรมต้องเรียงลำดับต่อกันจากน้อยไปมาก
- line number แรกสุดของโปรแกรมที่ถูก INCLUDE จะต้องมากกว่า line number ในประโยคที่มีคำสั่ง INCLUDE นั้นอยู่
- ช่วงของ line number ในโปรแกรมที่ถูก INCLUDE ต้องอยู่ก่อน line number ถัดไปในโปรแกรมที่มีคำสั่ง INCLUDE เช่น ถ้าในโปรแกรมหลักที่กำลัง compile เป็นดังนี้

```
100 % INCLUDE SUB1.BAS
200 IF A=1 THEN GOTO 500
```

โปรแกรม SUB1 BAS จะต้องมี line number อยู่ระหว่าง 101-199 แต่ข้อกำหนดเหล่านี้จะไม่จำเป็นในกรณีที่ compile โดยละการใช้ line number โดยใช้สวิทช์ /C ในตอน compile

- การใช้ INCLUDE จะใช้ซ้อนกันไม่ได้ส่งในโปรแกรมที่ถูก INCLUDE จะมีคำสั่งให้ไป INCLUDE โปรแกรมอื่นอีกไม่ได้
- คำสั่ง INCLUDE จะต้องเป็นคำสั่งสุดท้ายใน line ในกรณีที่ line นั้นมีหลายคำสั่งคั่นด้วย : เช่น

```
100 DEFINT I-N: % INCLUDE COMMON.BAS
```

11. ON ERROR GOTO และ RESUME

ถ้ามีการใช้ ON ERROR GOTO และ RESUME <line number> จะต้องใช้ /E ในการ compile และถ้ามีการใช้-RESUME,RESUME NEXT, RESUME O จะต้องใช้/X แทน/E การใช้สวิทซ์เหล่านี้ทำให้ file ที่เป็น .REL และ .COM มีขนาดใหญ่ขึ้น

12. REM ใช้เหมือนใน interpreter คือขึ้นต้นด้วย REM หรือ single quote(') จะบอกว่าประโยคนั้นเป็น Remark หรือหมายเหตุ compiler จะไม่แปลและไม่ทำให้โปรแกรมที่ได้จากการ compiler มีขนาดใหญ่ขึ้น

13. RUN ใช้ได้เฉพาะ RUN line number และ RUN ซึ่งจะเป็นการสั่ง RUN โปรแกรมที่ได้จากการ compile ภาษา BASIC หรือภาษาอื่น ๆ (file ที่เป็น COM) ไม่ใช่ BASIC source file ที่ต้อง RUN ด้วย interpreter ส่วน RUN<filename>, R ใช้ไม่ได้

14 STOP การทำงานเหมือนกับ END โดยปิดไฟล์ทั้งหมดและกลับสู่ CP/M เพียงแต่ไม่จำเป็นต้องอยู่ท้ายสุดของโปรแกรมอย่าง END ตามปกติเมื่อโปรแกรมหยุดด้วยคำสั่ง STOP จะมีการบอกแอดเดรสในรูปเลขฐานสิบหก ซึ่งเป็นตำแหน่งที่โปรแกรมหยุดลงออกมาด้วย และถ้าใช้ /D,/E หรือ/X ในตอน compiler ก็จะมีการบอก line number ที่หยุดออกมาด้วย



โปรดติดตามได้ในฉบับหน้าครับ Ø

โดย ชัยธวัช

ต่อทีวีเข้ากับไมโคร

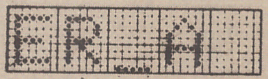
(ตอนที่ 2)

ในตอนที่แล้วเราได้อธิบายการทำงานของจอ CRT, ตัวกำเนิดอักษรบนจอภาพและหน่วยความจำสำหรับจอภาพ หวังว่าคงพอจะทราบหลักการกันบ้างแล้ว ขณะนี้เรายังไม่ทราบว่าจะจัดการกับข้อมูลบนจอภาพได้อย่างไร หรือถ้าจะต่อจอ CRT เข้ากับ CRT Controller เราต้องใช้สัญญาณต่าง ๆ อย่างไร ในตอนนี้เราจะอธิบายถึงการจัดการข้อมูลบนจอภาพ และปัญหาทางด้านสัญญาณต่างๆ ที่ต้องใช้ในการต่อ CRT Controller เข้ากับจอ CRT

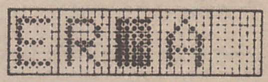
● การจัดการกับข้อมูลบนจอภาพ

เมื่อจอ CRT ถูกใช้เป็นส่วนหนึ่งของเทอร์มินัล (TERMINAL) สำหรับแสดงข้อมูลที่ใส่เข้าไปทางคีย์บอร์ด เราจำเป็นต้องมีวิธีจัดการและควบคุมตำแหน่งที่จะใส่ข้อมูลใหม่ลงบนจอภาพ โดยปกติแล้ว CRT Controller สามารถจะแสดง CURSOR บนจอภาพได้โดย CURSOR นี้จะทำหน้าที่ชี้ตำแหน่งที่จะใส่ตัวอักษรหรือข้อมูลเข้าไปบนจอและสามารถจะเลื่อนไปที่ใดบนจอภาพก็ได้ CURSOR โดยทั่วไปจะมีรูปร่างเป็นสี่เหลี่ยมทึบขนาดตัวอักษร หรือเส้นที่อยู่ใต้ตัวอักษรมีขนาดกว้างเท่ากับตัวอักษร อาจจะกะพริบหรือไม่กะพริบก็ได้แล้วแต่เราจะต้องการ ซึ่งก็ทำได้โดยการโปรแกรมเข้าไปที่ตัว CRT Controller

วิธีการบอกตำแหน่งที่จะใส่ข้อมูลบนจออีกวิธีคือ การใช้ปากกาแสง (LIGHT PEN) ซึ่งส่วนใหญ่ใช้ในงานทางด้านการฟฟิค เราเพียงแต่ถือปากกาจ่อหน้าจอที่ตำแหน่งที่ต้องการจะใส่ข้อมูลก็สามารถจะใส่ข้อมูลที่ตำแหน่งนั้นได้ วิธีการรับข้อมูลทางปากกาแสงนี้ต้องเพิ่มเติมวงจรพิเศษให้ทำงานร่วมกับ CRT Controller (CRT Cntroller บางเบอร์จะมีส่วนพิเศษนี้รวมอยู่ด้วย) ตัวปากกาแสงเป็นอุปกรณ์ที่สามารถจะรับรู้ตำแหน่งโดยใช้แสงซึ่งออกมาจากจอ CRT เมื่อลำอิเลคตรอนตกกระทบผิวจอ เมื่อรับแสงได้แล้วปากกาแสงนี้จะส่งสัญญาณไปที่ CRT Controller ซึ่งจะสามารถรู้ตำแหน่งปากกาบนจอภาพได้



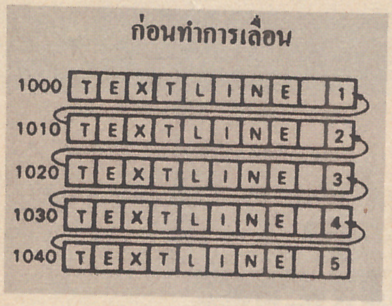
CURSOR แบบเส้นใต้

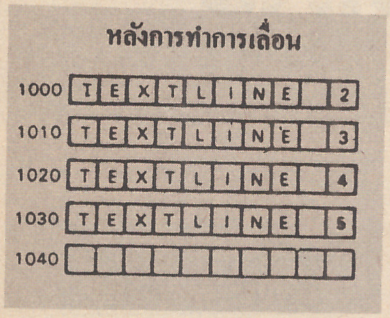


CURSOR แบบทึบ

ในกรณีที่จอ CRT ถูกใช้สำหรับแสดงข้อมูลที่เป็นข้อความจำนวนมาก ๆ (TEXT) ก็จำเป็นจะต้องมีการจัดการเกี่ยวกับจอเพื่อให้แสดงข้อมูลได้หมด โดยการทำให้จอภาพเหมือนกับเป็นกระดาษแผ่นยาว ๆ ข้อมูลจะถูกแสดงเริ่มตั้งแต่แถวอักษรบนสุดของจอภาพ ข้อมูลต่อมาก็จะถูกแสดงในแนวต่อ ๆ มา เมื่อถึงแถวล่างสุดของจอแล้วข้อมูลแถวบนก็จะถูกเลื่อน (SCROLL) ขึ้น เพื่อให้มีที่สำหรับแสดงข้อมูลในแถวต่อไปได้อีก การเลื่อนข้อมูลขึ้นจะทำให้อักษรแถวบนสุดหายไปจากจอภาพขณะที่เกิดแถวว่างด้านล่างเพิ่มขึ้นอีก

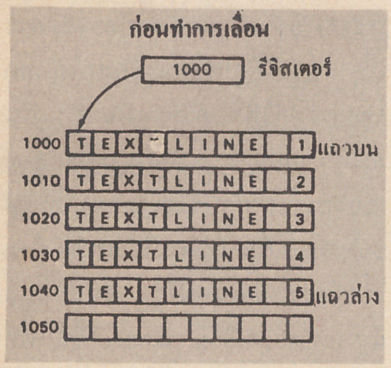
วิธีการเลื่อนข้อมูลมีด้วยกันหลายวิธี *วิธีที่ง่ายวิธีหนึ่งคือ การเลื่อนข้อมูลในหน่วยความจำจอภาพ* แต่วิธีการนี้ทำให้เสียเวลามากในการย้ายข้อมูลจากส่วนหนึ่งไปยังอีกส่วนหนึ่ง ซึ่งเป็นปัญหา

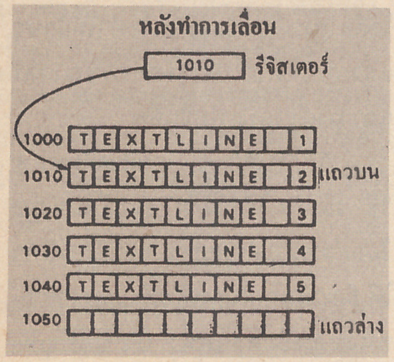




สำหรับไมโครโปรเซสเซอร์

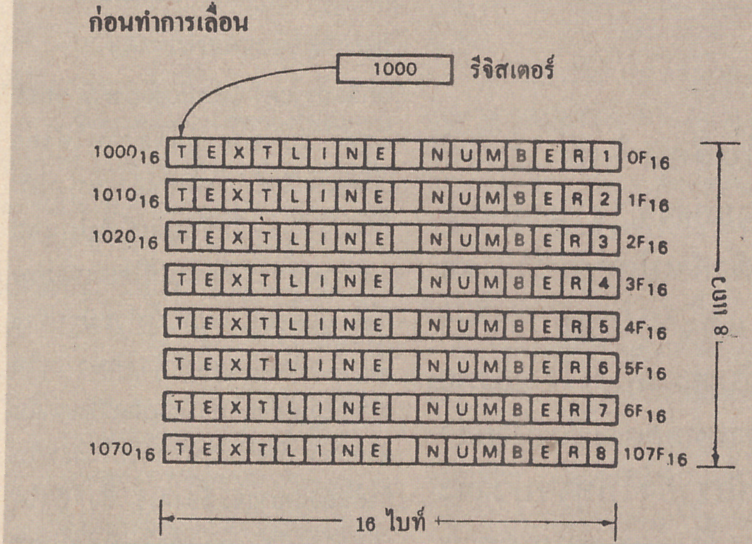
อีกวิธีหนึ่งถือการใช้ *รีจิสเตอร์* เก็บข้อมูลซึ่งชี้ตำแหน่งในหน่วยความจำที่เป็นส่วนบนสุดของจอภาพในขณะนั้น (ตำแหน่งในหน่วยความจำที่เริ่มต้นแสดงบนจอจะถูกเก็บไว้ในรีจิสเตอร์นี้)

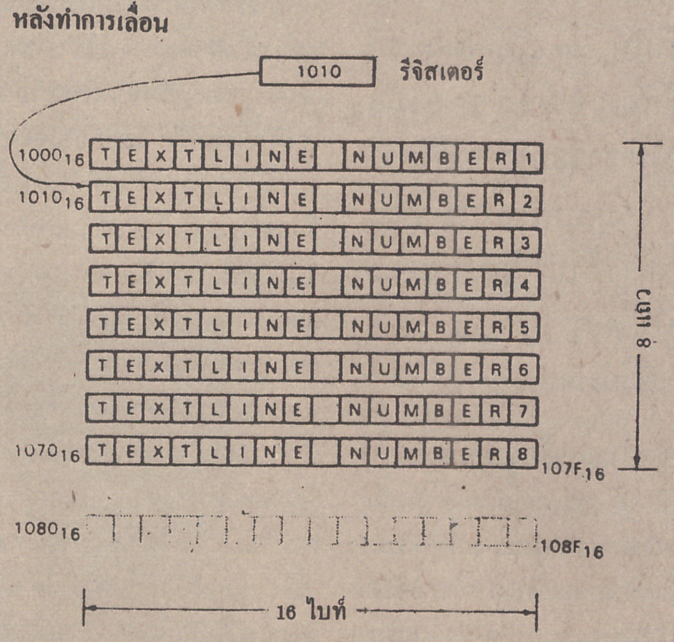




เมื่อต้องการจะเลื่อนข้อมูลที่จะแสดงก็เพียงแต่เปลี่ยนค่าในรีจิสเตอร์นี้ โดยการเพิ่มค่า ๆ หนึ่งเข้าไป (DISPLACEMENT) ค่านี้จะเป็นจำนวนตัวอักษรที่แสดงในแถวหนึ่ง ๆ โดยวิธีการนี้เราจึงไม่จำเป็นต้องเคลื่อนย้ายข้อมูลในหน่วยความจำจริง ๆ เราเพียงแต่เปลี่ยนตำแหน่งเริ่มต้นในหน่วยความจำนอกจากนี้เรายังสามารถเลื่อนข้อมูลลงได้อีกด้วย ทั้งนี้เพราะข้อมูลจริงก็ยังคงอยู่ที่เดิม

วิธีการนี้เราต้องแน่ใจว่าส่วนที่จะแสดงด้านล่างของจอภาพไม่มีข้อมูลอยู่ เพราะถ้ามีข้อมูลอยู่แล้วก็จะมาปรากฏบนจอด้วย ซึ่งเราคงไม่ต้องการเช่นนั้น เพื่อให้เห็นเป็นที่ว่างสำหรับใส่ข้อมูลใหม่ได้ จึงจำเป็นต้องใส่ข้อมูลที่แทนตัวอักษรว่าง (SPACE) เข้าไปที่หน่วยความจำส่วนนี้ กรณีที่เราเลื่อนจนกระทั่ง





แถวสุดท้ายที่เคยปรากฏหายไป เราต้องเปลี่ยนข้อมูลในรีจิสเตอร์เพื่อให้ไปชี้ที่ตำแหน่งเริ่มต้นอีกครั้งหนึ่งแทนที่จะเป็นตำแหน่งต่อจากแถวสุดท้ายซึ่งไม่มีจริงในหน่วยความจำ

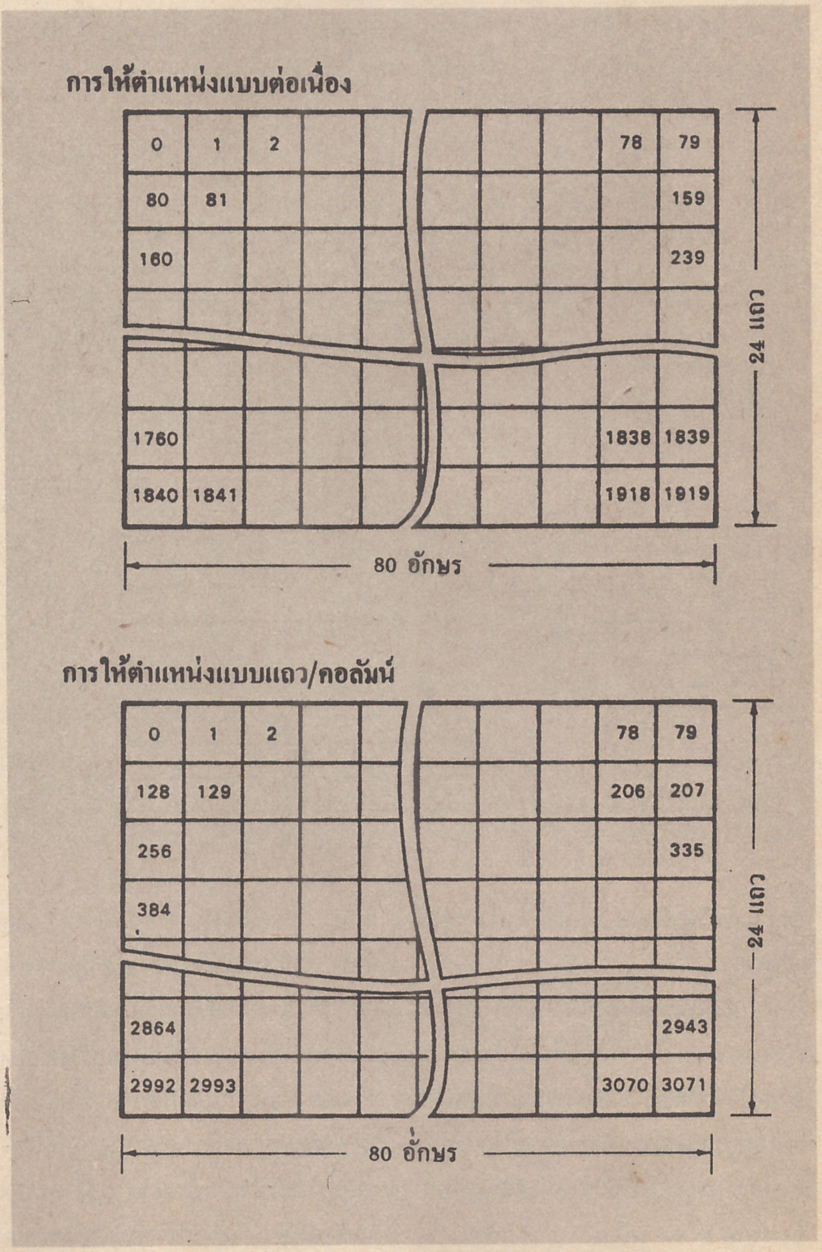
สำหรับการเปลี่ยนข้อมูลในรีจิสเตอร์ให้เป็นตำแหน่งเริ่มของหน่วยความจำอีกครั้งหนึ่งนั้นทำได้โดยการรีเซ็ทค่าในวงจรเลือกตำแหน่งของหน่วยความจำจอภาพให้เป็นตำแหน่งเริ่มต้นอีกครั้ง ซึ่งจะทำให้รีจีสเตอร์ที่เก็บข้อมูลเก็บตำแหน่งเริ่มต้นนี้ด้วยเช่นกัน

ที่เราได้กล่าวมาแล้วเราถือว่าตำแหน่งในหน่วยความจำจอภาพนั้นต่อเนื่องกัน วิธีนี้ไม่ใช่วิธีที่มีประสิทธิภาพเท่าใดนัก สมมุติว่าเราต้องการเพิ่มแถวหรือลบแถวออกหรือเพิ่มหรือลดตัวอักษรในแถวหนึ่งแถวใด เราจะไม่สามารถเข้าถึงตำแหน่งนั้นได้โดยตรง ทั้งนี้เพราะข้อมูลที่จะแสดงบนจอนั้นเราเก็บไว้ในหน่วยความจำแบบต่อเนื่องจากแถวหนึ่งไปยังอีกแถวหนึ่งต่อกันไป

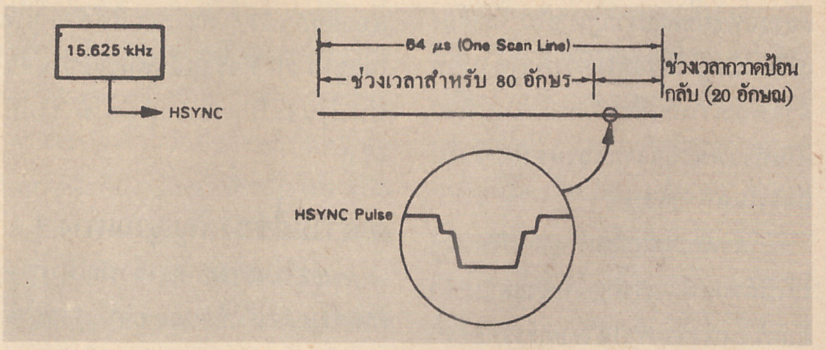
CRT Controller บางเบอร์มีการเลือกตำแหน่งในหน่วยความจำจอภาพเป็นแบบแถว/คอลัมน์ ซึ่งสัญญาณเลือกตำแหน่งในหน่วยความจำบางส่วนจะใช้เลือกแถวของตัวอักษรที่จะแสดง วิธีนี้จะลดความยุ่งยากในการจัดการข้อมูลที่จะแสดงบนจอลงได้แต่ในขณะเดียวกันก็จะลดประสิทธิภาพของหน่วยความจำลง การใช้สัญญาณเลือกตำแหน่งแบบต่อเนื่องตัวอักษรแรกในแต่ละแถวจะถูกเก็บต่อจากตัวอักษรสุดท้ายในแถวก่อนหน้านั้น ถ้าใช้สัญญาณเลือกแบบแถวคอลัมน์ ตำแหน่งเริ่มของอักษรแรกในแต่ละแถวจะไม่ต่อเนื่องกับตัวอักษรสุดท้ายในแถวก่อนหน้า วิธีนี้จะเหมือนกับว่าจำนวนอักษรในแต่ละแถว เป็นจำนวนเลขสองยกกำลัง (เพราะเราใช้สัญญาณเลขฐาน 2) ที่จริงแล้วหน่วยความจำบางส่วนจะไม่สามารถถูกใช้ได้ (ไม่มีจริง)

● ความถี่ของสัญญาณต่าง ๆ

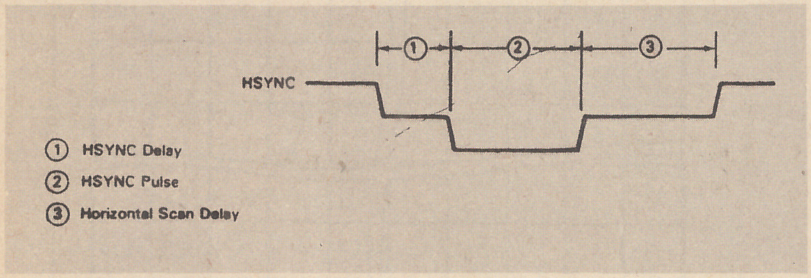
ต่อไปเราจะกล่าวถึงปัญหาทางด้านสัญญาณที่ใช้กับจอ CRT เราจะใช้ความถี่ของสัญญาณ HSYNC ที่ 15.625 KHz เป็นตัวอย่าง การกวาดเส้นทางแนวนอนแต่ละครั้งจะใช้เวลา 64 ไมโครวินาที (1 ÷ 15625) สัญญาณพัลส์ (PULSE) ของ HSYNC จะถูกส่งให้กับ CRT ในตอนที่การกวาดเส้นทางแนวนอนจะสิ้นสุดลงในแต่ละครั้งเวลาส่วนหนึ่งของ 64 ไมโครวินาทีนี้ จะถูกใช้สำหรับให้สัญญาณ พัลส์ HSYNC และสัญญาณสำหรับการกวาดป้อนกลับ โดยทั่วไปเวลาประมาณ 20 เปอร์เซ็นต์ของเวลาที่ใช้ในการกวาดทางแนวนอนทั้งหมดจะถูกใช้สำหรับการกวาดย้อนกลับนี้ ความสัมพันธ์ระหว่างส่วนที่ใช้กวาด,



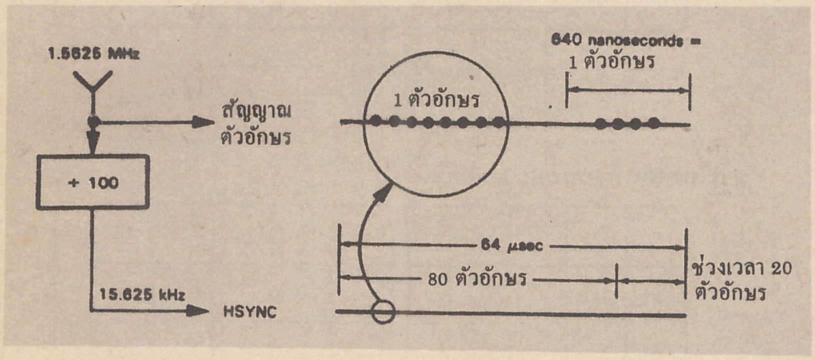
สัญญาณพัสล์ HSYNC และเวลาสำหรับการกวาดย้อนกลับแสดงดังรูป



นอกจากนี้สัญญาณการกวาดป้อนกลับทางแนวนอนยังสามารถแบ่งได้เป็น 3 ส่วนคือ สัญญาณหน่วง HSYNC, สัญญาณพัลส์ HSYNC และสัญญาณหน่วงของการกวาดทางแนวนอน (HSYNC DELAY, HSYNC PULSE, HORIZONAL SCAN DELAY) ซึ่งอธิบายได้ดังรูป



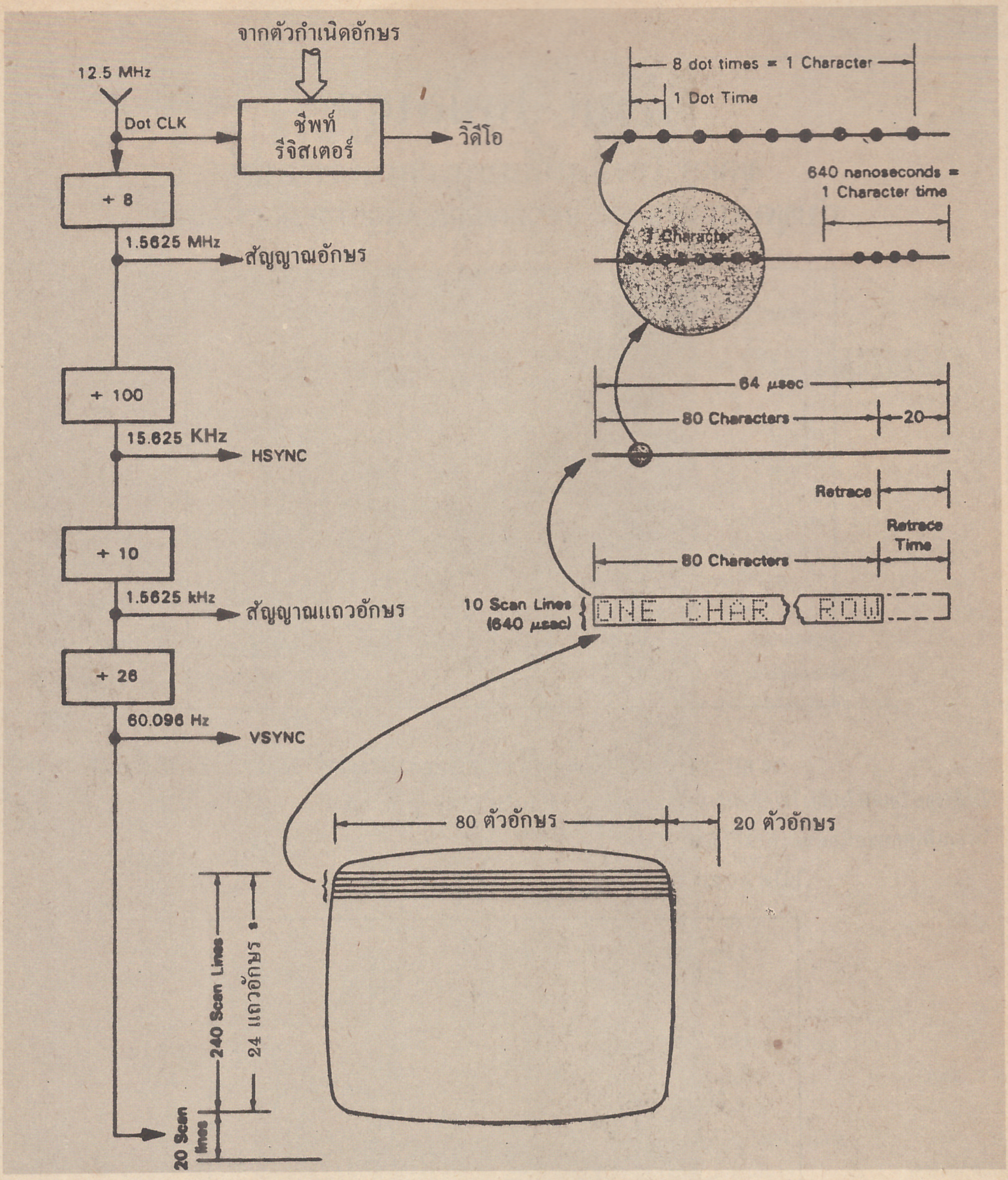
ถ้าเราต้องการแสดงตัวอักษร 80 ตัวใน 1 แถว เราจะต้องแบ่งเวลาที่ใช้ในการกวาดทางแนวนอนแต่ละครั้งออกเป็น 100 ส่วน เพื่อที่ว่าจะได้ให้อีก 20 ส่วน (20%) ของเวลานี้เป็นส่วนสำหรับการกวาดย้อนกลับ ดังนั้นเราจะใช้เวลาสำหรับตัวอักษร 1 ตัวใน 1 แถวเป็น 640 นาโนวินาที (64 ไมโครวินาที ÷ 100) ดังรูป



ดังนั้นเราต้องคูณความถี่สัญญาณ HSYNC ด้วย 100 เพื่อจะให้ได้สัญญาณสำหรับตัวอักษร 1 ตัว เพราะฉะนั้นจะได้สัญญาณสำหรับตัวอักษรเป็นความถี่ 1.5625 MHz ในที่นี้เราจะใช้ตัวอักษรขนาด 6 × 8 จุดใน เมทริกซ์ขนาด 8 × 10 เป็นตัวอย่างเพื่ออธิบาย ตัวอักษรแต่ละตัวในแถวจะต้องถูกแบ่งเวลาเป็น 8 ช่วง สัญญาณนาฬิกาสำหรับจุดซึ่งเราให้แก่ ชิฟท์รีจิสเตอร์ เพื่อให้กำเนิดสัญญาณวิดีโอ จะมีความถี่ 12.5 MHz ($1.5625 \times 10^6 \times 8$) เนื่องจากเราใช้เมทริกซ์ขนาด 8 × 10 แต่ละตัวอักษรจึงต้องกวาดเส้นทั้งหมด 10 ครั้ง หรือ 640 ไมโครวินาที ซึ่งได้รวมเวลาสำหรับการกวาดย้อนกลับทางแนวนอนไว้แล้ว เราจะได้สัญญาณสำหรับแถวตัวอักษรเป็นความถี่ 1.562 KHz (15.625 KHz ÷ 10)

ถ้าสัญญาณที่ใช้สำหรับ VSYNC มีความถี่ 60 Hz จากความถี่ที่เราใช้สำหรับ HSYNC 15.625 KHz เราต้องทำให้เป็นสัญญาความถี่ 60 Hz เพื่อใช้เป็นสัญญาณ VSYNC ในการกวาดย้อนกลับทางแนวตั้ง (เพื่อจะได้ไม่เกิดปัญหาที่เราได้กล่าวมาแล้ว) เราจะต้องหารสัญญาณความถี่ 15.625 KHz นี้ด้วย 260 ($15.625 \times 10^3 \div 260 = 60.096$ Hz) ดังนั้นเราจะได้เส้นกวาดบนจอทั้งหมด 260 เส้น ซึ่งจะได้อักษร 26 แถวแต่เวลาส่วนหนึ่งจะถูกใช้สำหรับการกวาดย้อนกลับทางแนวตั้งและเวลาที่ใช้ในการกวาดย้อนกลับทางแนวตั้งของ CRT โดยทั่วไปประมาณ 1 มิลลิวินาที ซึ่งเวลาของการกวาด 20 เส้นทางแนวนอนจะถูกใช้ไปเป็นเวลาสำหรับการกวาดย้อนกลับทางแนวตั้งนี้ ($20 \times 64 \times 10^6 = 1.28 \times 10^{-3}$) ได้นั้นจะเหลือเส้นกวาด 240 เส้น หรืออักษร 24 แถวบนจอ

จากที่กล่าวมาทั้งหมดเป็นการอธิบายการแสดงผลออกทางจอเป็นตัวอักษร สำหรับการแสดงผลเป็นรูปภาพนั้นเราจะต้องสามารถจัดการกับทุก ๆ จุด บนจอภาพได้ ซึ่งวิธีการ



ความถี่สำหรับสัญญาณต่าง ๆ

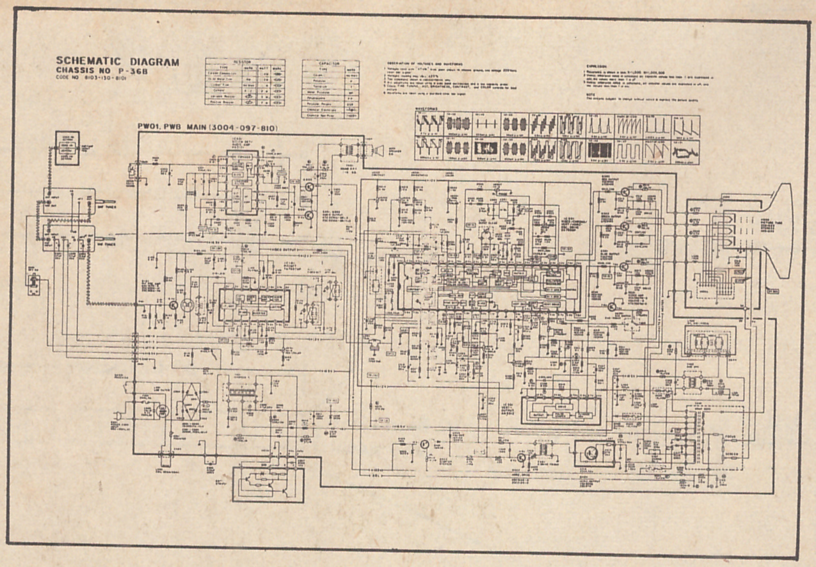
จะแตกต่างจากที่เราได้กล่าวมาแล้ว เช่น การเก็บจุดบนจอภาพจะต้องเก็บ 1 จุด ต่อ 1 บิท และข้อมูลจากหน่วยความจำก็จะถูกส่งทีละ 8 บิท ผ่านชิฟท์รีจิสเตอร์เพื่อเปลี่ยนเป็นข้อมูลอนุกรมส่งไปที่สัญญาณวีดีโอ หน่วยความจำที่ใช้ก็ย่อมต้องมีขนาดใหญ่ขึ้นตามความละเอียดของจุดบนจอภาพ เราได้อธิบายหลักการอย่างคร่าว ๆ ซึ่งใช้ในการแสดงผลเป็นตัวอักษรออกทางจอภาพ และหลักการทำงานของ CRT Controller มาแล้ว หวังว่าบทความนี้คงจะมีประโยชน์สำหรับผู้สนใจโดยทั่วไป สำหรับการนำ CRT Controller มาประยุกต์ใช้งานหรือสร้างวงจร INTERFACE CRT นั้นต้องมีการศึกษารายละเอียดและคุณสมบัติของ CRT Controller แต่ละเบอร์ซึ่งอาจจะยุ่งยากซับซ้อนกว่าที่กล่าวมาแล้ว

∅

ด่วน! ใหม่ล่าสุด

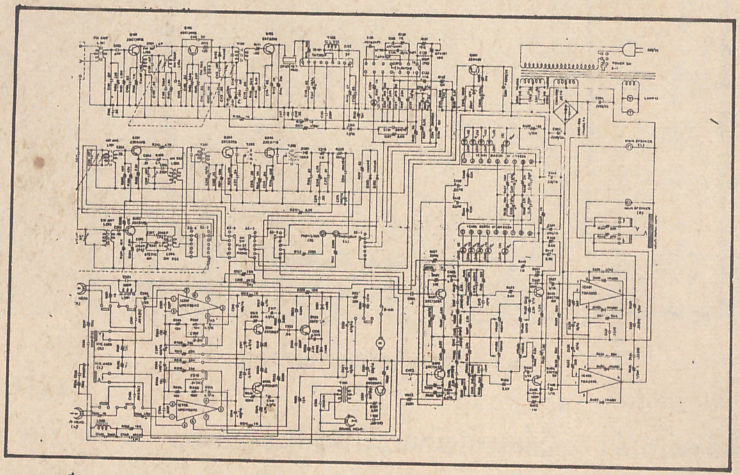
ชุดKIT โทรทัศน์สีสมบูรณ์แบบพร้อมตู้

กับชุด KIT STEREO RECEIVER/CASSETTE RECORDER



ชุด KIT โทรทัศน์สี 14 นิ้วและชุด KIT STEREO RECEIVER CASSETTE RECORDER *วงจรใหม่ ไม่ต้องจูนไอเอฟไม่ต้องใช้เครื่องมืออาไลเมนต์* เหมาะสำหรับไปประกอบทดลองเองหรือเพื่อการศึกษา และเสริมทักษะในงานอิเลคทรอนิคส์ของท่านอีกด้วย

ใช้ไฟ AC 220 V/DC 12 V



กรุณาติดต่อสอบถามรายละเอียด

TPR

บริษัท ทีวีพระโขนงเครื่องเย็น จำกัด

73/1 หมู่ 5 ถ.สุขาภิบาล 1 คลองกุ่ม บางกะปิ กรุงเทพ 10240

โทร. 377-9014-5

ตลาดนัดไมโครฯ

หน้านี้เรายกให้คุณฟรี!

ขายไมโครฯ NEC PC-8001 พร้อม DISK คู่ EPSON
สนใจติดต่อ **"อนุสรณ์"** โทร. 282-3691 หลัง 2 ทุ่ม

ต้องการขาย
SINCLAIR SPECTRUM 16 K, TAPE GAME 8 ม้วน, หนังสือ GAMES FOR ZX SPECTRUM, THE SPECTRUM GAMES COMPANION, คู่มือครบ รวมทั้งหมด 4,000 บาท (ใช้งานแล้ว 3 เดือน)
ติดต่อ **"ตั้ม"** 514-1021, เวลา 18.00-21.00 น.ทุกวัน

ขายเทปเกมส์ COMPUTER สำหรับเครื่อง APPLE หรือเครื่องเลียนแบบม้วนละ 40 บาท
สนใจขอรายละเอียดได้ที่ **สุทธิ์เภสัช** 1639/62 ถ.พหลโยธิน 15 สะพานควาย กทม. 10400 หรือ โทร. 2792368

ขาย All from U.K. (England)
1. Sinclair ZX81 (จาก U.K.)
2. Software Game (1K) & '0' Level Maths (16 K) (จาก U.K.)
3. I/0 port (Sound board expansion) (จาก U.K.)

รวมทั้งชุด ราคา 3,900 บ.
ติดต่อ **ศุทธินี** หรือ **ชัชวาล** โทร. 3928568, 3924415, 3922710

ขาย
1. ไมโครคอมพิวเตอร์ MPF II ของใหม่อุปกรณ์ครบ ราคา 5,000 บาท ขนาด 64 K
2. เทปเกมส์ของ SINCLAIR ZX SPECTRUM 15 ม้วน ๆ ละ 40 บาท

ติดต่อ**เงิน ฟองสินธุวานนท์** 1183/3 ซอยเดชา ยานนาวา กท. 10120 โทร. 2110078.

ขาย
CASSETTE INTERFACE FA-2 ของ CASIO ใช้แค่อาทิตย์เดียว 350 บาท เท่านั้น
ติดต่อ **ธีระ สุรรักษ์** 526/1 ซอยเพชรสมุทร 2 ต.แม่กลอง อ.เมือง จ.สมุทรสงคราม 75000

"ตลาดนัดไมโคร" เป็นเนื้อที่ที่ท่านจะสามารถใช้เป็นสื่อกลางในการติดต่อกับกลุ่มผู้อ่านไมโครอิเลคทรอนิคส์ ไม่ว่าจะเป็นการติดต่อแลกเปลี่ยนโปรแกรม ซื้อขายเครื่องเก่าหรือกิจกรรมอื่น ๆ ที่เป็นประโยชน์ด้านไมโครคอมพิวเตอร์เราจะพิจารณานำลงในหน้านี้ฟรี

กติกา กติกาคือข้อความที่ต้องการส่งมาลงต้องสั้นกระทัดรัด ไม่เกิน 60 คำ พร้อมกับส่งชื่อที่อยู่จริง พร้อมสำเนาบัตรประชาชนมาด้วย 1 ฉบับ

ส่งจดหมายมาที่ คอลัมน์ "ตลาดนัดไมโครฯ" วารสารไมโครอิเลคทรอนิคส์ บริษัทสยามขายน์ แอนด์เทคโนโลยี่ จำกัด 1091/230 ศูนย์การค้านครหลวง เพชรบุรีตัดใหม่ กรุงเทพฯ 10400 โทร : 252-2070

ดัชนีโฆษณา

ประกาศ

MICRO COMPUTER USER CLUB

ชมรมผู้ใช้ไมโครคอมพิวเตอร์
(ประเทศไทย)

รับสมัครนักเรียน นศ.ที่สนใจคอมพิวเตอร์เป็นจำนวนมากเพื่อจัดตั้ง **ชุมนุมคอมพิวเตอร์**

ทางชมรมจะเปิดอบรมวิชาการเขียนโปรแกรมด้วยภาษาเบสิค

- ทุกวันเสาร์ เริ่ม 10 มีนาคม 27
  16.00 - 19.00 น (APPLICATION IN BASIC)
- ทุกอังคาร, พฤหัส เริ่ม 15 มีนาคม 27
  16.00 - 17.30 น (BASIC)

ผู้สนใจติดต่อขอทราบรายละเอียดได้ที่ชมรม

236/5 ชั้น 3 โรงภาพยนตร์ลิโด ปทุมวัน กทม.10500

โทร. 2523695

ชมรมผู้ใช้ไมโครคอมพิวเตอร์
(ประเทศไทย)

โทร. 2523695

บอกข่าว

สมาชิกชมรมที่ประสงค์จะบอกข่าวการขายหรือฝากขาย ติดต่อได้ที่ชมรม

รายการคอมพิวเตอร์และอุปกรณ์ที่ฝากขายในเดือนนี้มีดังนี้

- PRINTER FX-100 สภาพเยี่ยมราคา 22,000.–
- CPU 68 Keys มี FUNCTION พร้อมจอภาพ 14 นิ้ว

ไม่สะท้อนจำนวนมาก ซึ่งต้องการขายในราคาถูก

236/5 ชั้น 3 โรงภาพยนตร์ลิโด ปทุมวัน กทม.10500

บริการใหม่ของชมรม

เพื่อความสะดวกและรวดเร็วของท่าน

- ชมรมได้เปิดศูนย์รวมไมโครคอมพิวเตอร์**เงินผ่อน**ขึ้น ซึ่งผู้สนใจสามารถซื้อได้ **เงินดาวน์ต่ำ ดอกเบี้ยน้อย ผ่อนนาน**
- สมาชิกชมรม เมื่อท่านนำบัตรสมาชิกไปแสดงที่ร้านหนังสือ **BOOK CHEST** สยามสแควร์ซอย 2 ท่านจะได้ส่วนลด **10%** และ **5%** ของหนังสือทุกชนิด
- วารสารฉบับที่ท่านถืออยู่นี้ **ขอรับได้ฟรี!** เมื่อท่านเป็นสมาชิกชมรม
- ชมรมมีโปรแกรม และเทปของเครื่อง SINCLAIR และ ATARI ไว้บริการแก่สมาชิกแล้ว
- เชิญชมวีดีโอ, เทป, โปรแกรมสำเร็จรูป

MICRO COMPUTER USER CLUB

ชมรมผู้ใช้ไมโครคอมพิวเตอร์
(ประเทศไทย)
236/5 ชั้น 3 โรงภาพยนตร์ลิโด ปทุมวัน กท.10500
โทร. 2523695

สมาชิก ใหม่

1. นายวิโรจน์ วิจิตรศรีพรกุล
2. ด.ช. สิทธิ สุธีวงศ์
3. ด.ช.สุรยุทธ์ เทือกสุบรรณ
4. นายทศพล สันติเทวกุล
5. นายทัศนพงศ์ หวังโภคา
6. นายบุญหลิบ พานิชชาติ
7. นายพิเชษฐชัย ศิริเยาวรัตน์
8. นายบุญภักดิ์ วงศ์วิรัติ
9. นายจุมพต จุนวีระพงศ์
10. นายสุภัทร์ หล่อเพ็ญศรี

MICRO COMPUTER USER CLUB

ชมรมผู้ใช้ไมโครคอมพิวเตอร์
(ประเทศไทย)

236/5 ชั้น 3 โรงภาพยนตร์ลิโด ปทุมวัน กทม.10500

AS-100

16-bit microcomputer

Canon

COMPUTER SYSTEMS

The tool for decision-makers



fma

Distribution by

FMA Corporation Ltd.

101 Sibunrueng 2 Building

1/7 Convent Road, Silom, Bangkok.

Tel. 235-0156-9, 235-0176-8